An Argus Specialist Publication

NOVEMBER 1984

80p

NEW

Your COMMODORE

YOUR BEST INDEPENDENT COMMODORE MAGAZINE



THE DIGITAL MUSICIAN

KEYBOARD CONTROL WITH THE MIDI INTERFACE

IMPROVE YOUR PROGRAMMING:
BASIC, MACHINE CODE, VIC GAMES AND BUSINESS ARTICLES INSIDE

As seen in the national press
WIN a DREAM HOLIDAY in our GREAT COMPETITION

VENTURE INSIDE FOR SOME ADVENTUROUS SOFTWARE

DOWN TO BUSINESS WITH THE SX-64: HARDWARE REVIEW INSIDE

Our COMMENT

Your trusty Editor has again tapped away at the typewriter to introduce this second fantastic issue of Your Commodore.

YOU KNOW IT ONLY seems a couple of months ago that we were sunning ourselves on the beach dreaming up the idea of a great magazine to cater for the needs of the Commodore user. And yet here we are in October looking forward to cosy roaring fires and sitting in front of our micros with the wind howling around outside the window! But there are all sorts of goodies on the Commodore scene to look forward to in the next couple of months.

What the future holds

At this very time Commodore are launching their new computers, the Plus 4 and the 16, Currah are announcing their speech synthesiser, Cheetah will be producing the Commodore version of their infra-red joystick (the RAT!) and the software companies will be starting their build-up to Christmas with the launch of undoubtedly numerous games and utilities for the Commodore range of machines.



The editorial team on your Commodore has been under intensive training to enable their finely tuned (?!) bodies and minds to keep up to date with all the latest happenings on the Commodore front, so all you need to do to ensure that you are kept informed is to make sure that you get your copies of Your Commodore regularly. There is an easy way to do this — just look for the Subscriptions page in this magazine, fill in the coupon, write your cheque and sit back and wait for your copies to come popping through your letterbox. It sure beats fighting your way to the shelves in your local newsagents!

Keeping in the present

This issue of Your Commodore we believe maintains the high standards set in the first issue: we have a review of the MIDI by Chris Palmer, who apart from being a bit of a whizz on computers is something of a talented musician; Runecaster has been brought up from the Crypt to tell us of Adventures and other things; we have pages packed with news and software reviews; and we have carried on our great series on machine code and BASIC. And, as if that isn't enough, we also have some fun games for you to type in and hints on how to become a 'sneaky programmer'!

Your views

It is always difficult when starting up a new magazine to gauge the response of the most important people involved — you, the readers. So here is your chance to get in on the act! By now we hope that you will have read the first issue of Your Commodore. The first question to be asked is: did you enjoy what you read? Then, was it useful/informative/fun? Did it tell you all you wanted to know?

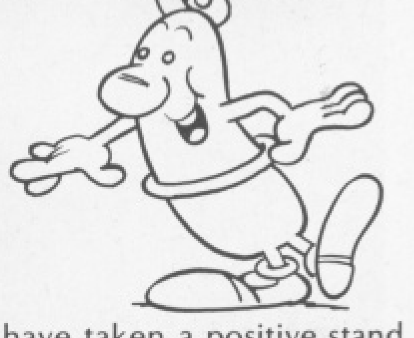
We consider ourselves fairly approachable here in the depths of the Your Commodore offices so why don't you use the lines of communication that we are trying to open up? Tell us what you would like to see in the future — would you like more games to type in, more programming features, less reviews, etc, etc.

Those good ol' lines of communication can also be used for getting into contact with other Commodre users; do you want to join a local Commodore users' group? Are you having trouble finding your way out of a particularly frustrating Adventure scenario? Having trouble finding just the right program to suit your specific needs?

All you have to do is write to the Editor at the London office and we'll do our best to help — either directly or by printing your letter within these hallowed pages.

ASP fights software piracy

Much has been said and written in condemnation of software piracy but few

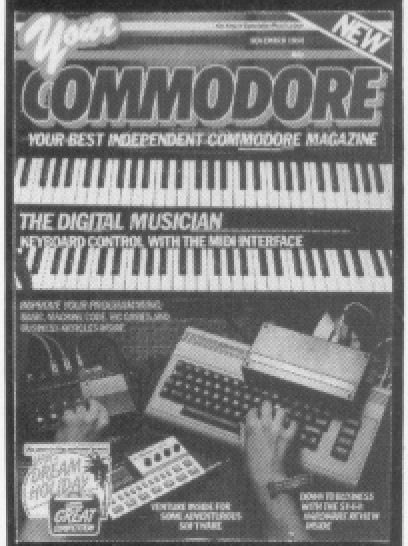


have taken a positive stand against it. ASP is among those few that have taken action to help curb the grave problem of home copying of commercial software.

ASP has already taken steps to eliminate advertisements in our magazines which relate to tape duplication for piracy purposes. While it is appreciated that individuals may take 'back-up' copies of their own programs, it should be noted that it is *ILLEGAL* to copy commercially available software for other than personal use.

Software piracy is costing the software industry huge sums of money which is detrimental to the future development of the industry. It is in everybody's interests to dramatically reduce the level of software piracy primarily because firms need funds raised from software sales to plough back intro research and development of new products. This means that the standard of software products can only improve.

ASP hopes our action will help combat this serious problem in order to maintain and improve the high standards of the UK software industry. We are asking *you* to do the same by refraining from duplicating or copying commercially available software for anything other than personal use.



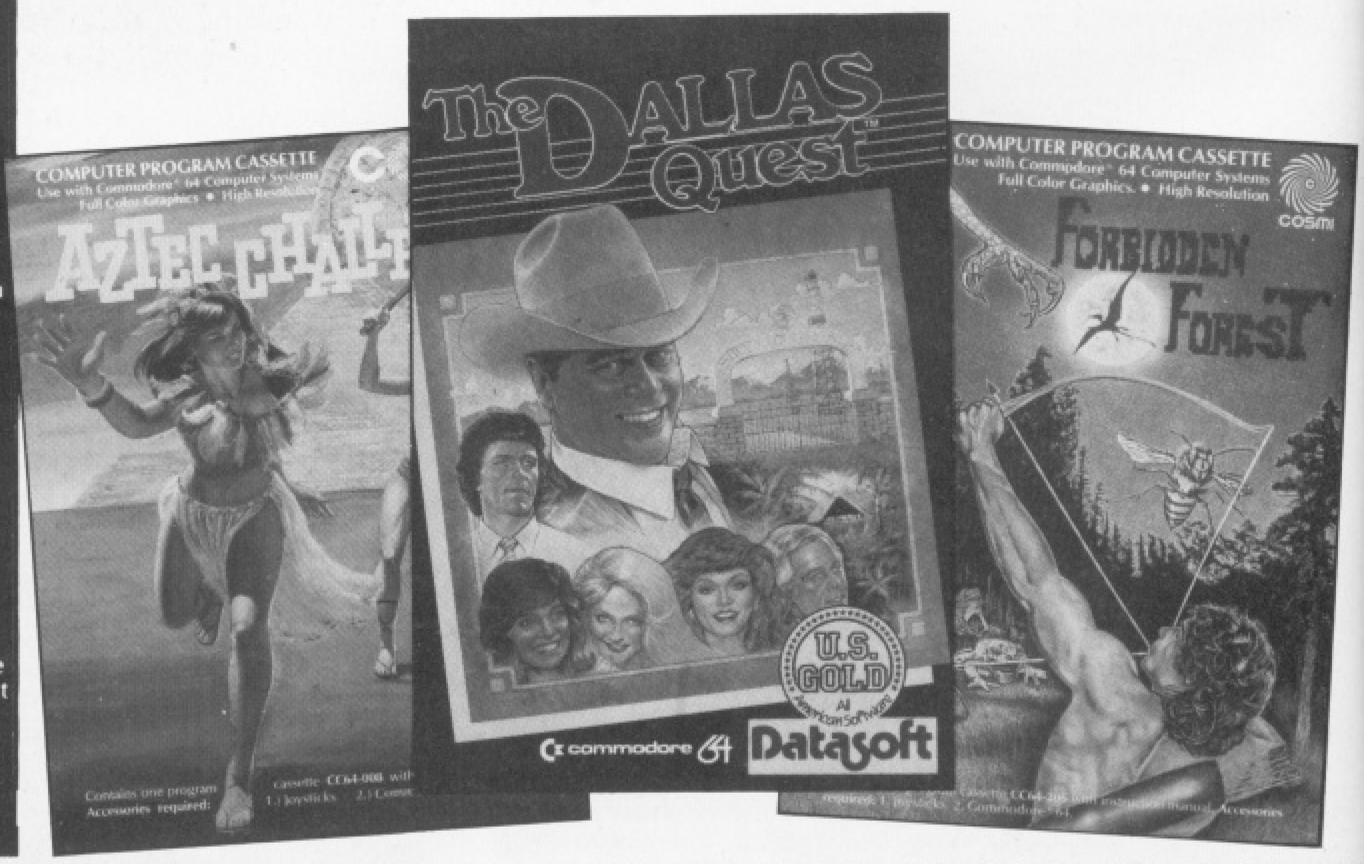
VOLUME 1
NUMBER 2
NOVEMBER 1984
Editor: Wendy J Palmer
Editorial Assistant: Alison Hjul
Advertisement Manager: Mike Segrue
Advertisement Copy Control: Sue Couchman
Chairman: Jim Connell
Origination: Ebony Typesetiing
Design: MM Design

Editorial & Advertisement Office
No 1 Golden Square,
London W1R 3AB
Telephone: 01-437 0626
Telex: 8811896

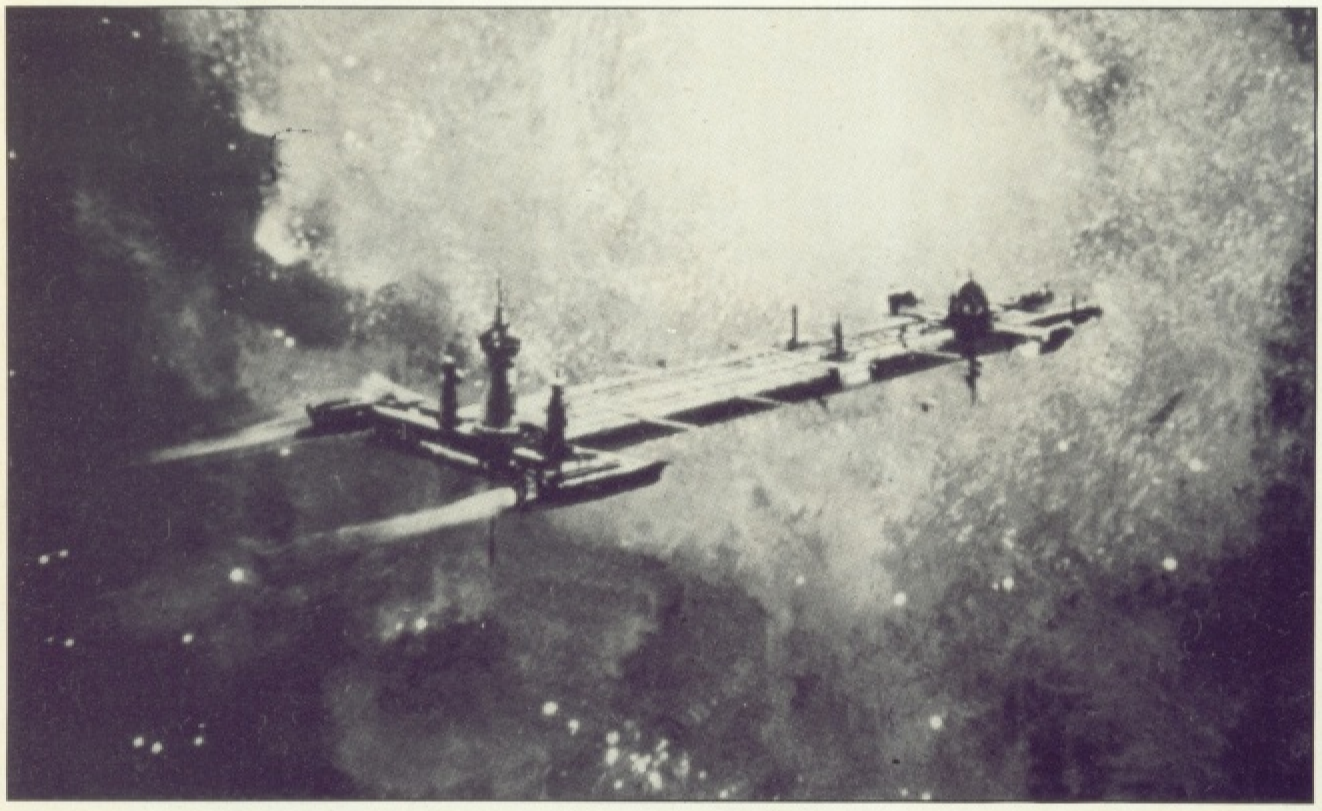
Your Commodore is a monthly magazine appearing on the first Friday of each month.

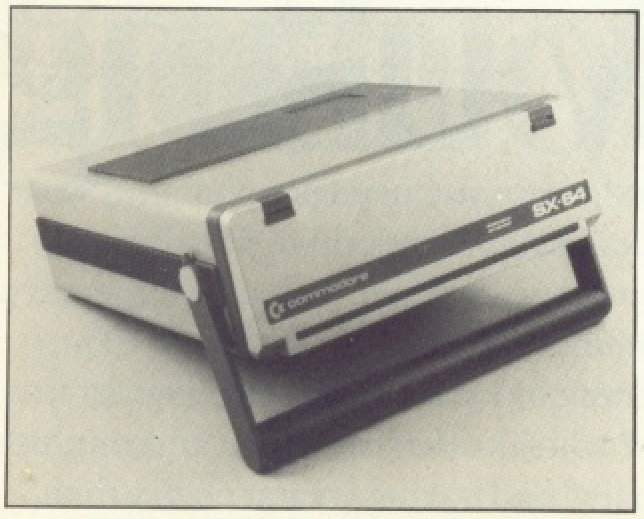
Distribution by: Argus Press Sales & Distribution Ltd, 12-18 Paul Street, London EC2A 4JS. Printed by: Alabaster Passmore & Sons Ltd, Tovil, Maidstone, Kent.

Subscription rates upon application to Your Commodore Subscriptions Department, Infonet Ltd, Times House, 179 The Marlowes, Hemel Hempstead, Herts. HP1 1BB.

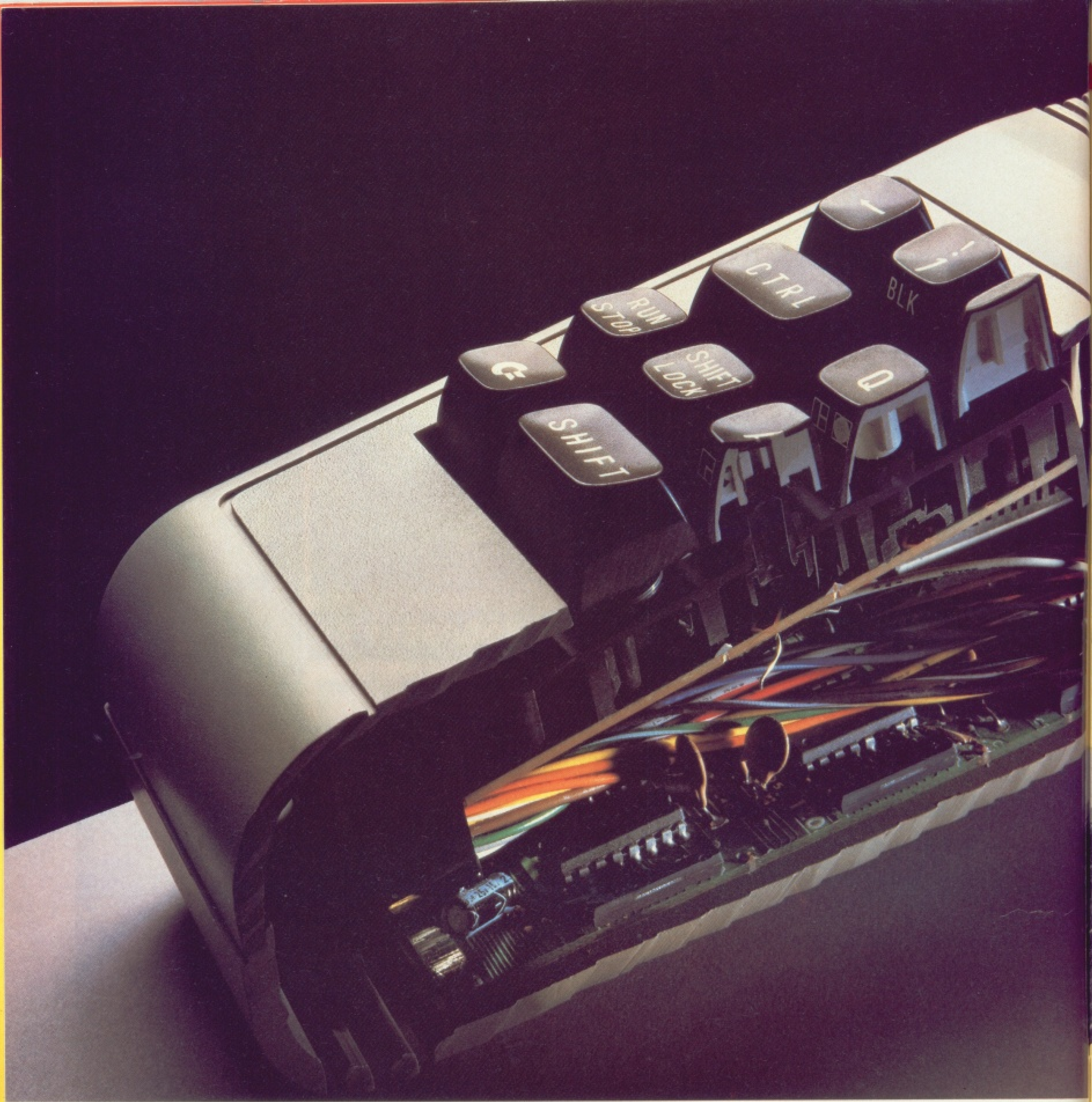


CONTENTS









Are you only using

To play only games on a Commodore computer is like asking Albert Einstein to work out the square root of four.

The computer's brain barely ticks over.

To really stretch it, you need more interesting software programs. For example, record keeping, interactive education, stimulating adventure games or word processing.

And for these you need peripherals.

Like a Commodore disk drive, a really fast storage and retrieval system with a vast memory.

Or a Commodore cassette unit, the inexpensive way of loading and storing programs.

For those who like the idea of text and graphics being more alive and having greater clarity than on a TV, there's the Commodore colour monitor.





COMMODORE MPS801 ☐
Dot matrix printer. £230.00. Tractor feed. Print speed: 50 characters per second.

COMMODORE MPS802 ☐
Dot matrix printer. £345.00. Friction feed for standard paper. Print speed: 60 characters per second.

COMMODORE DPS1101 ☐
Daisy wheel printer. £399.99. Letter quality print on all types of paper. Print speed: 18 characters per second.

COMMODORE 1520 ☐
Printer plotter. £169.99. For charts and graphs. Print speed: 14 characters per second.

COMMODORE 1541 ☐
Disk drive. £229.00. 170K memory. 5¼" diskette.

COMMODORE 1531 ☐
Cassette unit. £44.95. For Commodore 16 and Commodore plus/4.

COMMODORE 1530 ☐
£44.95. For Commodore 64.

COMMODORE 1701 ☐
Colour monitor. £230.00.

JOYSTICKS ☐
(prices from £7.50)

PADDLES (£13.50). ☐

Details correct at time of going to press.

g 1/10th of your brain?

And for hard copy, there are our three printers and a printer plotter. These will preserve on paper–in colour, black and white, chart form, graphs or text, the fruits of all your labour.

Finally, to make games playing more exciting, there are joysticks and paddles.

So use your brain. And make sure you use all of your computer's brain.

FOR FURTHER INFORMATION, TICK ONE (OR MORE) OF THE BOXES ABOVE AND SEND TO THE COMMODORE INFORMATION CENTRE, 1 HUNTERS ROAD, WELDON, CORBY, NORTHAMPTON NN17 1QX. TEL: CORBY (0536) 205252.

NAME ____________________

ADDRESS ____________________

____

PE YCO 1184

commodore

A.P. and D.J. Stephenson continue their layman's guide to machine code in part 2 of this series.

MASTERING MACHINE CODE

TO WRITE MACHINE code programs, it is important to know the space in memory which is free from the clutches of the operating system, the BASIC interpreter and the peripheral control area. This free space varies in different models. In the CBM 64, there is, fortunately, a healthy 4K of memory which is reserved for your own machine code programs. Machine code programs can be safely loaded into the 4K memory block starting from $C000 onwards. The character '$' will be used from now on to indicate where the number is in hex rather than decimal).

In addition to the space required to house the program, a need will arise for a few special memory locations in 'page zero' which is at the bottom of memory and extends from address $0000 to $00FF. In Part 1 of this series, we learned that the microprocessor communicates with the memory chips via a set of 16 wires called the **address bus** and a set of eight wires called the **data bus.** The memory may be considered as a huge block of separately addressable **locations**. Each location can hold eight bits and each location has a unique 'address' for identity purposes.

The binary pattern, which the microprocessor sends out on the address bus at any one time, energises one particular memory location. This pattern is the address. However, it is easier to think of the address in terms of hex rather than binary. Furthermore, it is conventional to consider the pattern on the 16-bit address bus in two halves. The eight most significant bits (A8 to A15) are known as the **page** and the eight least significant bits (A0 to A7) as the **address** on the **page.** It is also conventional to refer to the most significant half of the address bus as the **high byte** address and the least significant half as the **low byte** address. Refer to figure 2.1 which illustrates the concept of a page and an address on that page.

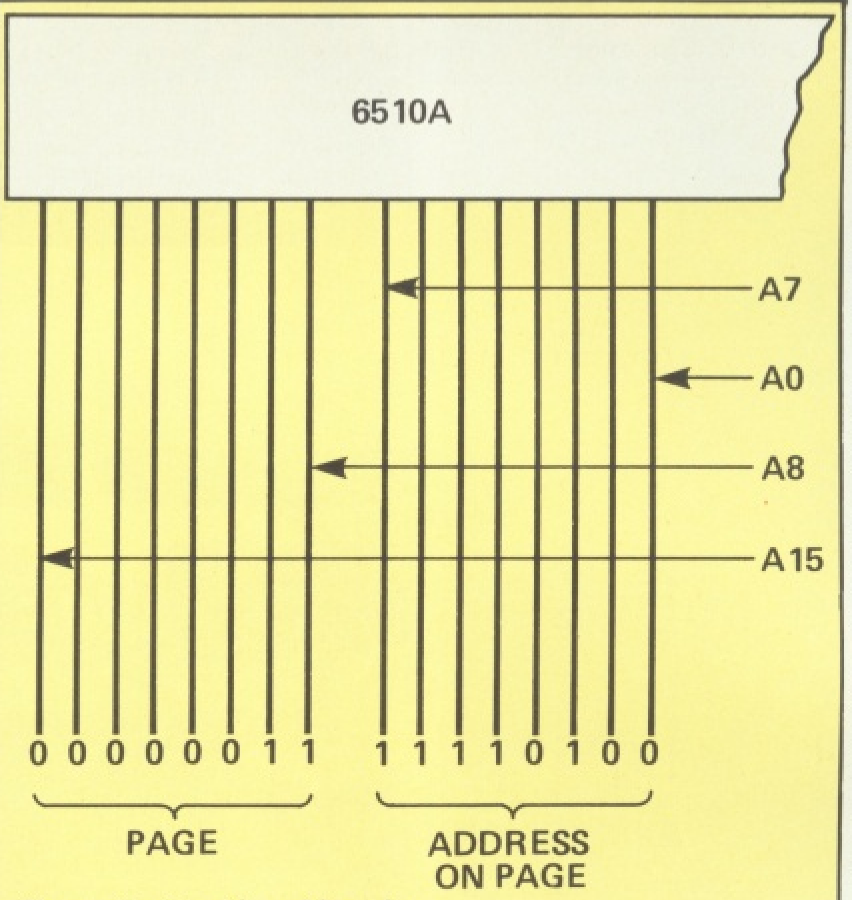


Figure 2.1 How the address bus should be visualised.

The example shows a sample binary pattern, 0000 0011 1111 0100 which, when translated into hex, becomes $03F4 (if you still cling on to decimal, this is 1012). Note that only four hex digits are required to express any of the 64K possible address combinations. Returning to the subject of pages and Figure 2.1, instead of saying the absolute address is $03F4, we could say the address is $F4 on page $03. We could drop the leading zero and simply say page 3. Before leaving the subject of pages, it is worth studying some of the figures involved in address work.

One page contains 256 addresses. In hex, the range extends from $00 to $FF. There are 256 pages in the complete memory map, so again, the hex range is from $00 to $FF. Check: 256 x 256 = 65,536 = 64K

If we have to write machine code without the aid of an assembler, we are forced to use decimal addresses because the CBM 64 does not cater for hex. Although brute force conversion from hex to decimal is quite in order, you are strongly recommended to keep in mind the division of the address bus into two sections. We should remember that a complete memory address occupies two bytes, the high byte for page and the low byte for address on the page. The high byte is worth 256 times as much as the equivalent low byte.

To choose a simple example, if the address is $0305 (address 5 on page 3), the decimal equivalent is 5 + (256x3) = 773.

Let's try the more difficult address, $250F, in order to practise some hex to decimal conversion. The low byte is $0F which in decimal is 15 and the high byte is $25 which is 37 decimal. So the complete address in decimal is 15 + (256 x 37) = 9487. If you intend to follow this series without obtaining an assembler, it will bring dividends if you spend a little time practising these methods of converting hex addresses to decimal.

The 6510 microprocessor

When you program in BASIC, the **microprocessor**, the workhorse of the computer, remains unseen in the background. There is no need to know what type it is, how many bits it can handle at once, how many registers there are inside it or what is the **repertoire of instructions.** The situation is different for the machine code programmer. The peculiarities of the resident microprocessor are all important.

The microprocessor used in the Commodore 64 is a 6510A. Readers who have been used to the well known 6502 microprocessor will be relieved to know that the two are software compatible. The only difference is that the 6510A has a few special output pins which the machine uses to control the cassette interface. It is possible to

plunge straight into machine code programming without troubling too much about the technical details of the 6510A. However, it pays dividends in the long run if some of the internal behaviour is understood and it can also be interesting for its own sake.

Programs written in machine code for any given microprocessor should, subject to minor variations, still run on any make of computer employing the same microprocessor. That is to say, machine code programs are microprocessor (rather than machine) specific. The 'minor variations' mentioned above include such things as differences in the way memory is allocated and the amount and location of free space. Machine code programs are usually written with the aid of an assembler and some variation in syntax can be expected between different commercial versions.

It is better to begin by reviewing the microprocessor in relation to other main components of the system. The microprocessor communicates with the rest of the computer via three bundles of wires known as 'buses'. As we have seen, the address bus is responsible for picking out the particular memory location required by the programmer. The data bus is responsible for sending or receiving data to and from the chosen location. The control bus is a hotch potch of wires, necessary for the overall discipline of the system.

The ROM chips

These contain fixed information and can not be subsequently altered by the computer. The information stored includes the 8K operating system of the computer (Commodore call this the 'kernal' ROM). The BASIC language interpreter is also an 8K ROM. The most important characteristic of ROMS is the permanence of the stored information which is retained after power is disconnected.

The RAM chips (Random Access Memory)

The title is misleading because the essential quality of RAMs, which distinguish them from ROMs, is the ability to change the stored information under program control. The mere fact that they are 'random' access is incidental because so also are ROMs. In other words, RAMS are really **read/write** memories. Depending on the internal structure, RAMS may be further classified into 'static' or 'dynamic'. Some writers refer to dynamic RAMS as DRAMs, the 'D' prefix standing for dynamic. Due to the need for reducing current consumption and maximising packing density, each bit is stored within the inter-electrode capacity of MOS transistors. The stored information, however, is a transient affair because it is only a minute electrostatic charge which leaks away in a few milliseconds. Consequently, each stored bit must be periodically re-charged in order to compensate for the leakage. This process, called 'refreshing', is inherent in the hardware design and is not the responsibility of the programmer. However, the refresh-cycle does take up extra time. Dynamic RAMs are therefore a compromise in which access time is sacrificed in order to increase packing density and reduce cost.

The CBM, and indeed nearly all other makes of microcomputers, will use dynamic RAMS. The alternative would be to use static RAMs but the cost would be prohibitive and they would occupy a greater space on circuit boards. From now, the term RAM will be taken to mean the dynamic type.

6510A systems are **memory mapped**, a term used to denote that peripherals are addressed as if they were ordinary memory locations.

Inside the 6510A

From the viewpoint of the programmer, the 6510A can be considered as a collection of **registers.** Each register can be considered as a separate memory location within the microprocessor. With one or two exceptions, all data must be fetched from memory via the data bus and routed to one or other of the registers before carrying out any operation. A machine code program will consist of a series of **instructions** which inform the microprocessor which registers are to be used to

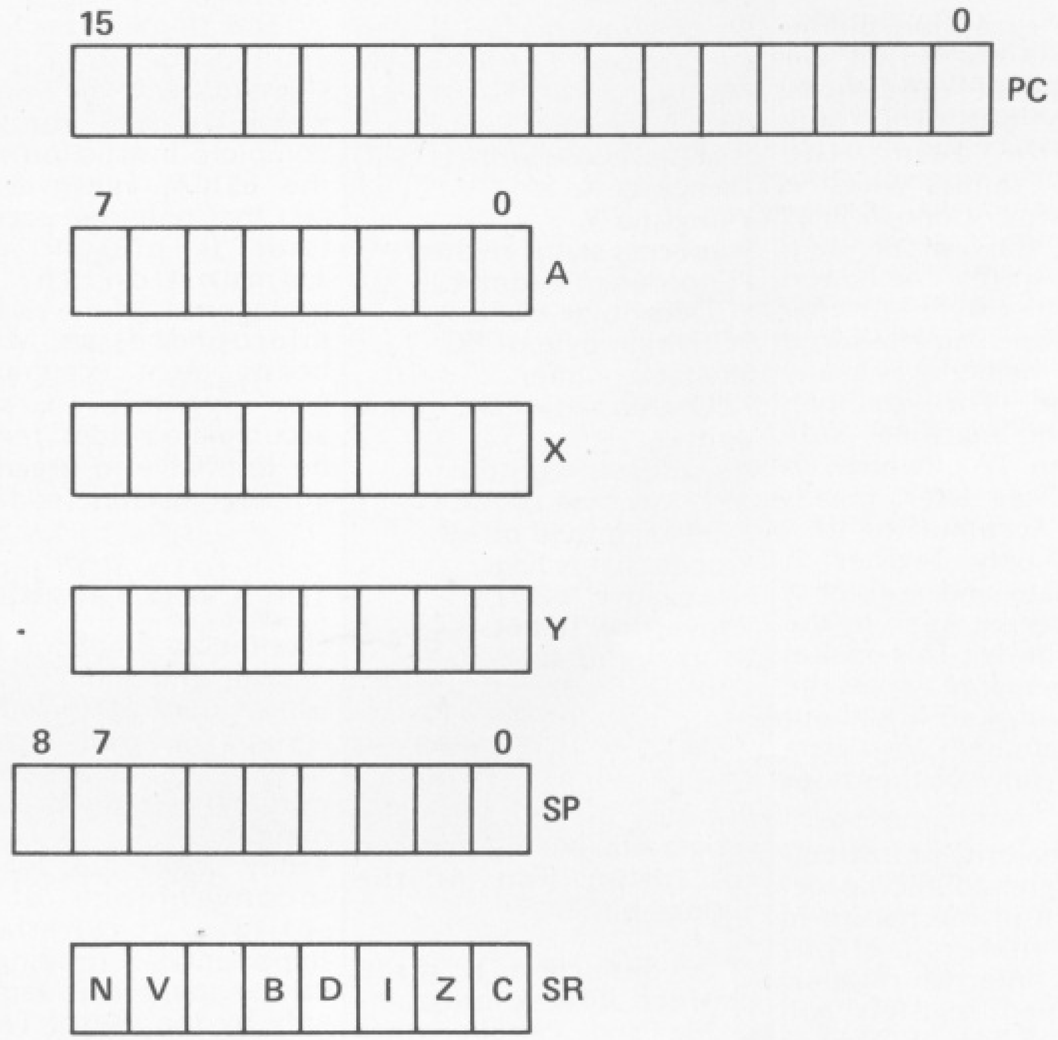


Fig. 2.2. The programmable registers and disposition of various bits

carry out the current task.

Registers do not have addresses, at least not in the same way as described above. If an assembler is used, they are called up by special code letters, such as A or X or Y etc which form part of each instruction. Paths, within the microprocessor, connecting the various registers together or to the external buses are sometimes called 'highways' because they ramify over the chip area like main trunk roads.

The registers

With one exception, all the registers in the 6510A are eight bits wide, the same as the data bus. The only exception being the Program Counter which is 16 bits wide. Control lines operate the input and output gates of each separate register, ensuring that only one pair is allowed access to the highway at any one time. For example, during the machine code instruction TAX (which, as we shall see later, means Transfer Accumulator to X register) only register A output gate and register X input gate are open to the data highway. This makes the highway free to pass the contents of A to X without being jammed by data resting in any of the other registers.

The majority of instructions we give to microprocessors are in the nature of data transfers, either between internal registers or between registers and external RAM, ROM or peripherals. Some instructions, such as ADC (ADd with Carry), perform arithmetical operations on the data but this may still have to be fetched from memory. Even a simple instruction like INX (INcrement contents of X) involves a transfer because the X register is not equipped for altering itself. Instead, the contents of X must be transferred along the highway to the arithmetic section before the 1 can be added.

Some commonly used abbreviations

Before we even attempt to write machine code programs or before even we can attempt the precise definition of a machine code instruction, we must understand the operation of the microprocessor registers. Certain bits in these registers have special significance according to the position they occupy. The following abbreviations and conventions are, more or less, standardised and will be used from now on:

1sb=least significant bit.
msb=most significant bit.
Bit positions within a byte are numbered 7 6 5 4 3 2 1 0.
Bit 0 is the lsb.
Bit 7 is the msb.
A=the accumulator.
X=register X.
Y=register Y.
P=process status register.
PC=program counter.
PCL=low byte of PC.
PCH=high byte of PC.
SP=stack pointer.
ALU=arithmetic and logic unit.
AR=address register.
ARL=low byte of AR.
ARH=high byte of AR.
Process status flags:
N=negative (bit 7).
V=overflow (bit 6).
B=break (bit 4).
D=BCD (bit 3).
I=interrupt (bit 2)
Z=zero (bit 1).
C=carry (bit 0).

Figure 2.2 shows the programmable registers and the disposition of the various bits.

A distinction is made between directly programmable and certain other registers which, although playing a vital role, remain in the background, unseen by the programmer. Instructions exist which allow the programmer to transfer date between memory and registers.

Accumulator (A)

this register has a supreme role. It is the only one capable of performing arithmetic processing. It is involved in transfers to and from memory and acts as interim data storage during arithmetic and logic operations. For example, during a simple addition of two numbers using the instruction ADC (ADd with Carry), the first number must pass to the accumulator and is then transferred to a temporary holding register within the ALU. The second number then enters A, the addition is carried out and the result sent back to A. The ALU in the 6510A, in common with nearly all other microprocessors, requires the two variables first, the add operator is then activated and the result passed to the accumulator, replacing the previous contents.

The dominance of the accumulator over other registers will be evident when we later study the complete instruction set of the 6510A. However, the fact that only one accumulator is present gives ammunition for the protagonists of the rival Z80 microprocessor which boasts eight accumulator type registers. A single accumulator does tend to be restrictive in organising efficient machine code.

The X and Y registers

Like the accumulator, the X register and the Y register (subsequently referred to as X and Y) are both eight bits wide. They have three primary uses in programming:

- They make up for the inconvenience of the solitary accumulator. Important data residing in A can be transferred temporarily by the use of TAX or TAY and later when A is free, transferred back using TXA or TYA.
- They can serve as up-counters or down-counters for setting up machine code loops. This is due to the ease by which they can be incremented or decremented by the instructions INX, DEX, INY or DEY. It is curious that the designers failed to provide an equivalent instruction for incrementing or decrementing A. The only way is by the relatively inefficient method of adding or subtracting 1, using ADC or SBC.
- They are fundamental to the technique known as **address modification by indexing.** When using an indexed addressing mode (denoted in assembly form by a comma followed by X or Y), the data in the X or Y register is automatically added to the given address. The resultant is interpreted as the final address of the required data.

This idea was pioneered by a team at Manchester University and, at the time, represented a step forward in computer science. They called the index register, the 'B box', presumably to differentiate it from the accumulator A. Prior to this, altering the address in loops was cumbersome. It involved loading the address part of an instruction from inside the program, incrementing it and then storing it back in the original position. In other words, it was necessary to alter the program in order to modify the address. Indexed addressing is so much cleaner to work with and certainly less error prone. Most of the indexable instructions in the 6510A allow a choice of using either X or Y indexing. Although indexed addressing is dealt with in some detail later, anticipation will do no harm, so study the following example:

Assume X contains 30 and we write LDA 100,X

The simple instruction LDA 130 would have the equivalent effect. They would both load the contents of address 130 into A. The advantage of the indexed over the simpler form will be apparent when organising loops involving action on consecutive addresses.

This should help to explain why the address bus, as well as the data bus, has access to the ALU. It should be understandable, if we realise that the index register contents have to be *added* to the operand. After all, the address modification

by indexing produces a computed address and only the ALU can truly compute.

The process status register (P)

If we define a register as an internal memory location for holding or processing data, then the Process Status register (P) is not a register at all. It is in fact a collection of isolated single-bit storage cells (flip-flops). Each bit is called a 'flag' because it conveys certain information in yes/no form either for the benefit of the machine or the programmer. The flags play an important role in the 'branch/if' type of instructions, the machine code is equivalent to the IF/THEN statement in BASIC. After most instructions, the relevant flags are updated, depending on the result they give. It is important for the programmer to understand the exact significance of each flag, that is to say, under what conditions they are set or reset. It is also important to know which are under sole control of the microprocessor and which are directly programmable.

```
10 033C        !###################
20 033C        !#          OLD KE
30 033C        !#
40 033C        !#      VERSION 1.0
50 033C        !#
60 033C        !# COPYRIGHT (C)
70 033C        !#
80 033C        !###################
90 033C        !
100 033C       !
110 033C       *=$C5C3
120 C5C3       !
130 C5C3       !
140 C5C3
```

```
150 C5C3              !VARIABLES AND EQU
160 C5C3              !
170 C5C3              LNKPTR       =   $A
180 C5C3              FNDLIN       =   $A
190 C5C3              PRTSTG       =   $A
200 C5C3              !
210 C5C3 E000         OLD          CPX #$
220 C5C5 F003                      BEQ DO
230 C5C7 4C08AF                    JMP $A
240 C5CA A901         DOOLD        LDA #$
250 C5CC A8                        TAY
260 C5CD 912B                      STA ($
270 C5CF 2033A5                    JSR LN
280 C5D2 A9FF                      LDA #$
```

```
290 C5D4 8514                      STA $1
300 C5D6 8515                      STA $1
310 C5D8 2013A6                    JSR FN
320 C5DB A902                      LDA #$
330 C5DD 18                        CLC
340 C5DE 655F                      ADC $5
350 C5E0 852D                      STA $2
360 C5E2 852F                      STA $2
370 C5E4 8531                      STA $3
380 C5E6 A900                      LDA #$
390 C5E8 6560                      ADC $6
400 C5EA 852E                      STA $2
410 C5EC 8530                      STA $3
420 C5EE 8532                      STA $3
```

The N Bit

If this is 1, the last result contained a 1 in bit 7 position. The N bit is often misleadingly called the 'sign bit' because two's complement arithmetic recognises bit 7 as the sign rather than magnitude. If the number is unsigned binary, the N flag merely indicates the state of bit 7. It is automatically set or reset and is not directly programmable.

The V bit

If this bit is 1, it indicates that the last instruction resulted in two's complement overflow, that is to say, the resultant number was too large to fit into a single byte. The programmer will always have the choice of working in unsigned binary or in two's complement form. If unsigned binary is used, the status of the V bit has no importance. The V bit is also used to indicate the status of bit 6 of a data byte when using a special instruction known as the BIT test. It is possible to clear the V bit to zero by the CLV instruction although there is no corresponding instruction to set it to 1.

The B bit

This is set to 1 when a BRK (break) instruction is encountered. It cannot be directly programmed.

The D bit

The 6510A normally uses straightforward binary arithmetic but it is capable of performing arithmetic on the assumption that all data is to be interpreted as BCD (Binary Coded Decimal). To force the use of BCD, the programmer must set the D bit to 1D by the instruction sED (SEt Decimal). This state remains until the instruction CLD (Clear Decimal) is used. BCD is not used very often so we will leave the details until a later part of this series.

The I bit

This is called the interrupt mask or the inhibit enable. If this bit is set to 1 by use of SEI (SEt Interrupt), all interrupt requests are refused until it is set to zero by CLI (Clear Interrupt). However, there is one special kind of interrupt instruction, NMI (Non Maskable Interrupt), which cannot be inhibited. The subject of interrupt is involved so it would be out of place to deal with at this stage.

The C bit

This is the Carry bit and is set to 1 when a carry out from the msb is detectd. Instead of the bit being pushed out at the far end and lost, it is 'caught' and placed in the C bit. At times, the programmer will consider it as the 'ninth bit'. It can provide a kind of bit-continuity between one byte and another. This is the essential idea behind mutiprecision

work where two or more bytes are used, connected end to end, to hold one number.

The Z bit

This bit is automatically set to 1 if the last instruction gave a zero result. It is easy to interpret this back to front so it is worth emphasising. If result=0, Z becomes 1. If result is non zero, Z becomes 0. It is used by the branch instructions BNE (Branch if Non Zero) and BEQ (Branch if Equal to Zero).

The stack pointer (SP)

This is an eight-bit register dedicated to the automatic control of a special area in page one of RAM defined as 'The Stack'. It is difficult to describe the action of the stack pointer without describing the stack so we must be content at the moment with the following brief description:

(1) The contents of SP is interpreted by the microprocessor as the address of the **currently vacant** location on the stack.

(2) To ensure that the address is always on page one, rather than page zero, a permanent 1 is hardwired at the msb end of SP to act as a ninth bit. If for example, SP contains 0000 0111, which is $07, the address is interpreted as 1 0000 0111 which is $107. That is to say, the address is $07 on page one.

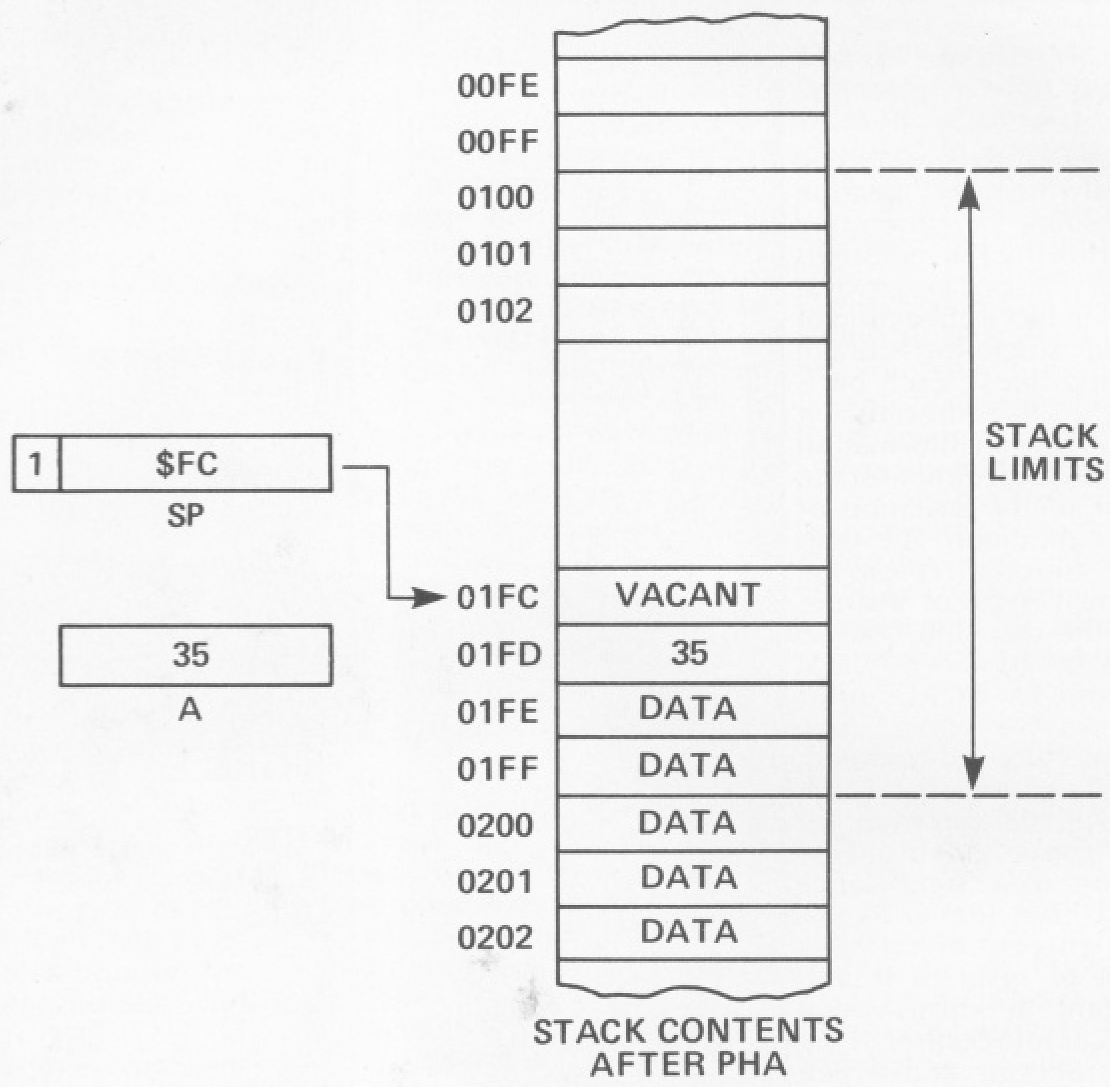
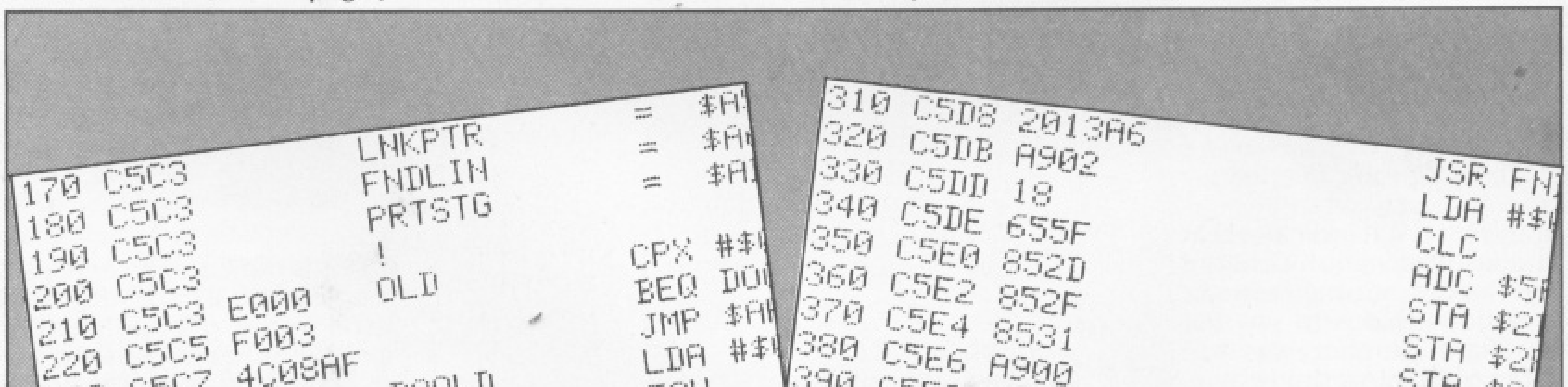


Figure 2.3 How the stack operates

(3) Special instructions exist for handling the stack, the two main ones being PHA (Push Accumulator) and PLA (Pull Accumulator). PHA will transfer A to the next available location on the stack and **decrements** SP so that it points to the next available location. PLA operates in the reverse manner. It first **increments** SP so that it points back again to the last valid entry and then pulls the contents of the stack location back to A. It may be evident from this brief desciption that data must be pulled back from the stack in inverse order. That is to say, the stack operates as a Last In First Out memory. In fact it is known as a LIFO memory stack. Figure 2.3 may help in visualising the stack.




ANIROG
ZAGA
MISSION
This diagonally scrolling maze game features superb 3D graphics brilliant sound effects and requires 100% concentration to successfully manoeuvre your helicopter through unknown hazards in order to complete Zaga Mission and live to play another day — Commodore 64 — £7.95
Also available on Disk at £9.95
TRADE ENQUIRIES: ANIROG SOFTWARE LTD. 29 WEST HILL DARTFORD KENT (0322) 92513/8
MAIL ORDER: 8 HIGH STREET HORLEY SURREY 24 HOUR CREDIT CARD SALES HORLEY (02934) 6083
PAYMENT BY CHEQUE P.O. ACCESS/VISA 50p POSTAGE & PACKAGING

STUCK?

DR WATSON series

Try a Course prescribed by Dr Watson

BASIC Courses In Space For The Commodore 64 & VIC 20

An exciting course for all children from 7 to 77. Learn BASIC while enjoying this most exciting space adventure. Master the computer to escape the Alien Spaceship.

It combines a story which is compulsive reading for the young reader with a carefully structured approach to BASIC intricately interwoven into the text. Also, each BASIC command covered is given a separate, careful explanation in a special 'easy reference' section in the second half of the book.

The tape includes the programs on the ship's computer, and four computer aided learning (CAL) programs which teach the major BASIC commands.

"It should hold the interest of an under-12 very well."

– Home Computing Weekly.



BOOK & TAPE

£9.95

Beginner's BASIC For The Commodore 64

This project-based course guides you step-by-step through BASIC, developing games and utilities using progressively more complex BASIC commands. All the programs developed are provided on tape so that you can get a taste of them first, before working along with the detailed explanations given in the book.

Programs include three video ball games, a character generator utility, a sprite generator utility and a 'composatune' music utility. You also get 'Honey.Aid1', a machine-code utility that adds 28 commands to Commodore 64 BASIC allowing easier prgramming, sound and graphics.



BOOK & TAPE

£10.50

"The Dr Watson series is rapidly becoming recognised as quality material."

"... a well thought out tutorial ... if this was the total content of the course, it would prove excellent value. However, there is more ... much more."

" ... the course consistently encourages you to learn more and more by making learning fun."

"other ... manufacturers have been shown how an introductory manual should be written."

– Commodore Computing International.

Beginner's Assembly Language Courses For The Commodore 64, VIC 20 And PET

These courses introduce the real beginner to assembly language programming. No prior knowledge of assembly language is assumed and the aim is to ensure that *every* reader succeeds. Numerous examples illustrate the points while exercises along with solutions test the understanding.

The tape includes an assembler which assembles and disassembles code anywhere in memory. Also on the tape is a binary/BCD/hexidecimal CAL tutor program which teaches about the various mathematical notations used in machine code programming.



BOOK & TAPE

£12.50

"There may be easier ways of adding two and two together ... But there aren't easier ways of learning how your computer does it."

– PCN.

"No self-respecting VIC 20 owner interested in programming should miss this instruction set."

"The manual would be worth buying for the reference section alone."

"Don't miss it".

– Which Micro.

"I have made more progress in a week than messing around for a year with other books and articles."

"a really super clear book ... "

– The VIC 20 North London User Group.

" .. this book is worth its weight in gold."

"If there was ever a good beginner's guide in this field, then this is it."

– Commodore Computing International.

HONEYFOLD

HONEYFOLD SOFTWARE LTD
B1 Standfast House
Bath Place
High Street, Barnet
London EN5 1ED
Tel: 01-441 4130

A FREE LIGHTPEN

Yes a FREE Light Pen! with every **GRAPHKIT**! Graphkit is the ultimate in drawing!, designing! or painting! (using the light pen of course, which incidentally has a full 3 year warranty!) Graphkit will amongst other things allow you to:

- Draw free handed on the screen anything from a Micky Mouse to your latest integrated circuit! With all the fill colour and erase facilities of course!
- Save/load your latest master piece to/from the disk/tape drive
- Or directly copy the drawing on the screen to the printer! And keep it for ever or send to a friend as a post card!
- Design your own colourful sprites and characters
- Play the games in the package! or use the light pen in your own games/education/programs

GRAPHKIT is available on disk and tape. Tape version is £1595 and disk version (recommended) is £19.95.



All you 1540/1541 disk users **DMON**! Is here at last. Dmon is the disk monitor you have been waiting for it will allow you to:

- read/write blocks from/to the disk
- display and edit blocks on the screen
- display and send disk messages/commands
- transfer your programs from tape to disk or disk to disk or even disk to tape! • and more!

DMON comes on DISK at only £9.95.

Do you use tapes? Are you sick and tired of waiting for your programs to be Loaded/Saved from/on the tape? Then you need a **TORNADO**!! Tornado allows you to Save/Load/Verify your Basic/machine code programs faster than a CBM 1541 disk drive does! Due to popular demand! Tornado now comes with new and more powerful commands plus extra instructions to assist you in making fast versions of your existing machine code/Basic programs. Tornado is available on tape for CBM 64 and 8K + Vic 20.

Do your Run/Stop and Restore keys often fail? Do you want to come out of those crashes?! Or get into those unbreakable programs?! Then what are you waiting for, get yourself a **BREAKER**!! Reset switch and let your computer know who is the boss!! Breaker can be connected to your machine in seconds, no soldering. Included with the Breaker is a copy of basic recovery software on tape. Now available for any CBM 64 or Vic 20.

MAD SUMMER OFFER A FREE BREAKER WITH EVERY TORNADO IF YOU USE THE COUPON BELOW

BSF CSL, 82 FURNACE DRIVE, CRAWLEY, W SUSSEX RH10 6JE.

Please send me ....Tornado at £9.95 + my free! Breaker

Please send me ....Breaker at £7.95

Please send me ....Rom-File at £7.95

Please send me ....Graphkit at £15.95 (tape)

Please send me ....Graphkit at £19.95 (disk)

Please send me ....DMON at £9.95

I enclose cheque/PO/cash to the sum of £..................

Name...........................Computer...........................

Address...

....

Please use this form and block letters only. All prices are inclusive of p&p (mail order only, add £2 if outside UK). Please post to:
BSF CSL, 82 FURNACE DRIVE, CRAWLEY, WEST SUSSEX RH10 6JE.


MIND GAMES
AMERICAN FOOTBALL
A fast & furious graphics strategic simulation
£9.99
FREE RULE BOOK
FOR THE SPECTRUM 48K, CBM 64
OTHER GREAT MIND GAMES:
MIND GAMES
QUEST FOR ETERNITY
Starring The Overlords of the Universe
The candidate (you) have to get to the Chamber of Creation. It's a laugh a minute, since it's 2,000 light years away on the most horrible planet in the Universe ... and your starship doesn't work either!
MIND GAMES
ORION QUEST
A full feature adventure starring well known nasty aliens the Zarps. Can you play the hero and stop their plans to blow up the earth.
MIND GAMES
STAR FORCE SEVEN
Starring The Zurgs
After a desperate space battle only one fleet of heroes remain to prevent the invasion of earth. The future of humanity lies with you!
Argus Press Software Group
For mail order, write with cheque/P.O./card No. to:
Mind Games, Argus Press Software Group, No.1 Golden Square, London W1A 3AB.

In part two of his series on BASIC games programming for VIC 20 users, B.M. Phillips gets things under control with PEEKs, POKEs and moving about.

VIC GAMES PROGRAMMING

THIS IS THE SECOND OF A five part series of BASIC games programming for the VIC 20. The series is primarily intended for newcomers to games programming, but there might well be a few useful tips for seasoned programmers.

As you will have discovered last month there can be a lot of satisfaction in creating a unique screen design, but it's static, non-interactive, and you could have used paper and pen to achieve the same result! What you are really doing when you design a screen display is to create an arena, whether it's for a space battle, gunfight scenario, or mowing a lawn. Let's have a look at getting it all moving.

If you enter Listing 1 and run it you will have a screen,

Listing 1:

```
1 REM INITIAL DISPLAY
2 :
10 CH=42:CL=2
20 P1=8164:P2=38884
100 PRINT"♥"
110 FORY=0TO22STEP22
120 FORX=0TO21
130 POKEP1+X-22*Y,160
140 POKEP2+X-22*Y,0
150 NEXTX,Y
160 FORY=1TO21
170 FORX=0TO21STEP21
180 POKEP1+X-22*Y,160
190 POKEP2+X-22*Y,0
200 NEXTX,Y
300 X=10:Y=11
310 POKEP1+X-22*Y,CH
320 POKEP2+X-22*Y,CL
330 GOTO330
```

surrounded by a boundary, containing a star in the middle. There are two ways of making the star move. Either interactively or under program control. Most games programs contain both elements. Firstly we'll consider moving the star interactively, in other words press the keys or move the joystick and the star moves.

The screen can be regarded as a 22 x 23 matrix with the bottom left hand corner being designated 0,0. Characters (CH) and colours (CL) can be POKEd onto the screen using:

POKE P1+X-22*Y,CH
POKE P2+X-22*Y,CL

where P1=8164

P2=38884

and all you have to do to put the character anywhere on the screen is to specify X and Y. You also have to rub out it's previous location, which you do simply by POKEing a space.

So far so good. All that remains is to get the information from you into the computer. There are lots of ways of doing this — from the keyboard, paddles, joystick, lightpen, microphone (if you have the appropriate add-on) etc. The most widely used methods are the KEYS and JOYSTICK, and I'll start off by describing the two most common methods of Key Input.

The first means of input is GET A$. You can see how this works if you add lines 330-500 (Listing 2) to the first program. The sequence of events is quite straightforward, but there's a problem. When you press "I" the star moves up the screen. OK that's what we want. Carry on pressing it. Lost it! Where's it gone? Now you're in dangerous waters. Your POKEing around in memory locations is best left alone. STOP!!! You'll crash the

Listing 2:

```
1 REM KEY INPUT
2 :
330 X1=X:Y1=Y
340 POKE650,128
397 :
398 REM PROGRAM LOOP
399 :
400 GETA$
410 IFA$=""THEN500
420 IFA$="I"THENY=Y+1
430 IFA$="M"THENY=Y-1
440 IFA$="J"THENX=X-1
450 IFA$="L"THENX=X+1
460 POKEP1+X1-22*Y1,32
470 POKEP1+X-22*Y,CH
480 POKEP2+X-22*Y,CL
490 X1=X:Y1=Y
500 GOTO400
```

computer! That was an example of bad programming. For it to work it relied on you stopping at the screen boundary. One of the first rules in Games programming is not to rely on the player — they always let you down. Instead, you make it fool proof.

There are two easy ways of doing this. The first is to look at the value of X and Y and if they are outside of your required range, either define the offending co-ordinate to the other end of the screen (wrap around)

i.e. 10 IF X > 21 THEN X=0
20 IF X < 0 THEN X=21

or stop it moving

i.e. 10 IF X< 0 OR IF X > 21 THEN X=X1

If you modify lines 420-450 as shown in Listing 3, the star will stay on the screen. You are now in control of the action. Why not have a go at modifying it to give the wrap around affect.

Listing 3:

```
1 REM NUMERICAL SCREEN CONTROL
2 :
420 IFA$="I"THENY=Y+1:IFY>21THENY=21
430 IFA$="M"THENY=Y-1:IFY<1THENY=1
440 IFA$="J"THENX=X-1:IFX<1THENX=1
450 IFA$="L"THENX=X+1:IFX>20THENX=20
READY.
```

The other way round the problem is to get the star to "look" where it's going, by PEEKing the location before moving, and if it's OK carry on, and if not stop dead. This technique is shown by adding lines 400-520 (listing 4) to the first program which will prevent the star moving unless there's a blank space available. This is the better method in general, as it is more flexible, and that PEEK can do far more than keep it

Listing 4:

```
1 REM PEEK SCREEN CONTROL
2 :
400 GETA$
410 IFA$=""THEN520
420 IFA$="I"THENY=Y+1
430 IFA$="M"THENY=Y-1
440 IFA$="J"THENX=X-1
450 IFA$="L"THENX=X+1
460 PK=PEEK(P1+X-22*Y)
470 IFPK=160THENX=X1:Y=Y1
480 POKEP1+X1-22*Y1,32
490 POKEP1+X-22*Y,CH
500 POKEP2+X-22*Y,CL
510 X1=X:Y1=Y
520 GOTO400
```

on the screen. You could use it to initiate an explosion, eat a monster, or squash a frog. PEEKs are pretty useful.

You might have noticed in running these programs that a subtle change has come over the keyboard. All the keys now auto-repeat. This was achieved by POKE650,128 (line 340). If you want to turn this off POKE650,0.

Another method of getting input from the

keyboard is to PEEK(197). This has the same effect in practice as the GETA$ statement, but the VIC handles it in a different way and it's faster. If you want to try this method out, you'll need to know the value of PEEK(197) for the relevant keys. You can do this by running the short program given in listing 5. For now though, we'll stick to GETA$.

Listing 5:

```
1 REM FIND PEEK(197)
2 :
10 PRINT"[CLR]"
20 GETA$:IFA$=""THEN20
30 PRINTA$,PEEK(197)
40 GOTO20
```

Getting to grips with the joystick

Now you know how to get input from the keys, the joystick will present no problem. The joystick contains 5 switches, four for movement and one for fire. All you have to do is find out which one is closed, and then continue as for the keys. There are a number of ways of doing this and one of the easiest is that described in the "Programmer's Reference Guide". I've adapted this for use in our program below (Listing 6. Lines 30-50 initialise the joystick, and the subroutine (lines 1000-1020) reads the values, and updates X and Y accordingly).

It's easy to get characters moving around the screen under program control. Again you use the X,Y co-ordinate idea and update X and Y each time you go round the program loop. A good way of doing this is to use:

X=X+DX
Y=Y+DY

The type of movement you get depends on the values of DX and DY. These can be continuously re-calculated as the program progresses to give, for example, a projectile (Newton's Laws of Motion), or made to change if the object hits something. This is demonstrated in BOUNCE (Listing 7) where a ball bounces around the screen. O.K. — boring stuff, but it's the basis of Break Out or Video Tennis, and maybe you could do something with it.

Listing 6:

```
1 REM JOYSTICK CONTROL
2 :
10 CH=42:CL=2
20 P1=8164:P2=38884
30 DIMJS(2,2):POKE37139,0:DD=37154:PA=37137:PB=37152
40 FORI=0TO2:FORJ=0TO2:READJS(J,I):NEXTJ,I
50 DATA-23,-22,-21,-1,0,1,21,22,23
100 PRINT"[CLR]"
110 FORY=0TO22STEP22
120 FORX=0TO21
130 POKEP1+X-22*Y,160
140 POKEP2+X-22*Y,0
150 NEXTX,Y
160 FORY=1TO21
170 FORX=0TO21STEP21
180 POKEP1+X-22*Y,160
190 POKEP2+X-22*Y,0
200 NEXTX,Y
300 X=10:Y=11
310 POKEP1+X-22*Y,CH
320 POKEP2+X-22*Y,CL
330 X1=X:Y1=Y
340 POKE650,128
397 :
398 REM PROGRAM LOOP
399 :
400 GOSUB1000
410 IFLT=0ANDRT=0ANDUP=0ANDDN=0ANDFR=0THEN520
420 IFRT<>0THENX=X+1
430 IFLT<>0THENX=X-1
440 IFUP<>0THENY=Y+1
450 IFDN<>0THENY=Y-1
460 PK=PEEK(P1+X-22*Y)
470 IFPK=160THENX=X1:Y=Y1
480 POKEP1+X1-22*Y1,32
490 POKEP1+X-22*Y,CH
500 POKEP2+X-22*Y,CL
510 X1=X:Y1=Y
520 GOTO400
997 :
998 REM JOYSTICK SUBROUTINE
999 :
1000 POKEDD,127:RT=-((PEEK(PB)AND128)=0):POKEDD,255
1010 P=PEEK(PA):DN=-((PAND8)=0):LT=((PAND16)=0):UP=
1020 FR=-((PAND32)=0):RETURN                    ((PAND4)=0)
```

Listing 7:

```
1 REM BOUNCE
2 :
10 P1=8164:P2=38884
20 CH=81:CL=2
30 X=10:Y=0:X1=10:Y1=Y
40 DX=1:DY=1
50 PRINT"[CLR]"
100 X=X+DX:Y=Y+DY
110 IFX<1ORX>20THENDX=DX*1
120 IFY<1ORY>21THENDY=DY*1
130 POKEP1+X1-22*Y1,32
140 POKEP1+X-22*Y,CH
150 POKEP2+X-22*Y,CL
160 X1=X:Y1=Y
170 GOTO100
```

Before we get on with the Game — yes there's a game in here somewhere — it's worth thinking about using the PRINT statement to move things about. It's a shame that the VIC hasn't got a PRINT AT command, it would make life easier. However, you can use PRINT to move things. Type in and RUN listing 8. A rocket descends. Now, that rocket is made up of 16 characters. Have a go at writing a program which POKEs those characters on and off the screen. It won't be as good — there are too many characters. You could use this idea for the final stages of a Lunar Lander program. Start off with the rocket descending as a single character, and when it gets near to the ground switch to a magnified version of the rocket controlling the display with PRINT.

Listing 8:

```
1 REM PRINT MOVE
2 :
100 W$="[graphics characters]"
110 B$="[graphics characters]"
120 U$="[graphics characters]"
130 PRINT"[CLR]"
140 FORI=1TO14
150 PRINTTAB(8)W$;
160 PRINTTAB(8)U$
170 FORJ=1TO500:NEXTJ
180 PRINTTAB(8)B$
190 PRINTTAB(8)U$
200 NEXTI
210 PRINTTAB(8)W$
220 GOTO220
```

Eat your heart out

Now for the game. In the last article I suggested that you could use a random maze as the basis of a game. The program in Listing 9 is an example. You move a snake around a random, shifting maze, eating hearts. One point per heart, and if 5 hearts are on the screen at any one time the game ends. I've written the game for key (I,J,K, and L) input, but if you don't like these keys change them. Better still, if you've got a joystick, incorporate that in the program.

Up to now we've moved a single character around the screen, but here we're moving six. Actually we only *appear* to be moving six. All you have to do to move a snake is to move the head (HD), fill in the space left with a body section (BD), and blank out the tail. This will give you smooth movement, and works well even with quite long snakes. You have to re-define the position of each body section each time you go round the loop, and this is done in lines 530-560. The snake "looks" where it is going, and if it meets itself, the maze, or the boundary, it stops (line 470), and if it meets a heart it eats it and the score (SC) goes up by 1. The number of hearts generated is totalled (TL), and the number of hearts on the screen at any time is given by TL-SC (line 810).

As it stands the game is playable, but in need of improvement. Presentation is important, and it is untidy. There should be some instructions, a replay facility, a high score record, and a much neater screen display particularly regarding the scoring. But then it's not finished — have a go and see what you can do with it. While you're at it think about the memory

used. This little program takes up 1200+ bytes. If you can cut this down you'll get an improvement in speed, and have more memory over for the frills. You can easily get it down to less than 1K by missing out REMS, using multiple statement lines, subroutines, DEF FNA's etc. If you can get below 900 bytes you're doing well.

By now you should have come up with a reasonable game. The only problem is that it's all a bit quiet — where's the sound? I'll go into that in the next article.

Listing 9:

```
1 REM SNAKE MAZE
2 :
10 BD=42:HD=90:C1=2:C2=6
20 P1=8164:P2=38884
100 PRINT"■"
110 FORY=0TO22STEP22
120 FORX=0TO21
130 POKEP1+X-22*Y,160
140 POKEP2+X-22*Y,0
150 NEXTX,Y
160 FORY=1TO21
170 FORX=0TO21STEP21
180 POKEP1+X-22*Y,160
190 POKEP2+X-22*Y,0
200 NEXTX,Y
300 X=10:Y=11:X(1)=X:Y(1)=Y
307 :
308 REM POKE HEAD
309 :
310 POKEP1+X-22*Y,HD
320 POKEP2+X-22*Y,C1
327 :
328 REM POKE BODY
329 :
330 FORI=1TO5
340 POKEP1+(X-I)-22*Y,BD
350 POKEP2+(X-I)-22*Y,C1
360 X(I+1)=X-I:Y(I+1)=Y
370 NEXTI
390 POKE650,128
397 :
398 REM PROGRAM LOOP
399 :
400 GETA$
410 IFA$=""THEN570
420 IFA$="I"THENY=Y+1
430 IFA$="M"THENY=Y-1
440 IFA$="J"THENX=X-1
450 IFA$="L"THENX=X+1
460 PK=PEEK(P1+X-22*Y)
465 IFPK=83THENGOSUB700:GOTO480
470 IFPK<>32THENX=X(1):Y=Y(1):GOTO400
477 :
478 REM MOVE SNAKE
479 :
480 POKEP1+X-22*Y,HD
490 POKEP2+X-22*Y,C1
500 POKEP1+X(1)-22*Y(1),BD
510 POKEP1+X(6)-22*Y(6),32
527 :
528 REM RE-DEFINE BODY
529 :
530 FORJ=6TO2STEP-1
540 X(J)=X(J-1):Y(J)=Y(J-1)
550 NEXTJ
560 X(1)=X:Y(1)=Y
567 :
568 REM RANDOM MAZE
569 :
570 RX=INT(RND(1)*18)+2
580 RY=INT(RND(1)*19)+2
590 RN=102:RL=INT(RND(1)*100):IFRL>95THENRN=83
600 CR=P1+RX-22*RY
610 IFRL<50THEN630
620 IFPEEK(CR)=32THENPOKECR,RN:POKECR+P2-P1,C2:GOSUB800:GOTO640
630 IFPEEK(CR)=102THENPOKECR,32
640 GOTO400
697 :
698 REM UPDATE SCORE
699 :
700 SC=SC+1:PRINT"■"SC
710 RETURN
797 :
798 REM COUNT HEARTS
799 :
800 IFRN=83THENTL=TL+1
810 IFTL-SC>4THENPRINT"■■■■GAME OVER":END
820 RETURN
```



Your COMMODORE

YOUR BEST INDEPENDENT COMMODORE MAGAZINE

SO YOU OWN A COMMODORE?

SO YOU'VE WRITTEN SOME PROGRAMS?

SO WHY HAVEN'T YOU SUBMITTED THEM TO US?

Your Commodore is always on the lookout for new material for publication and we know that there are thousands of intelligent, literate, innovative and creative Commodore owners out there, so why don't we get together?

If you have written an exhilirating game or an invaluable utility on your Commodore micro, share your talents with us and our readers by submitting your efforts and the form to the address below. All articles should be documented and type-written and should be accompanied by a printout of the program as well as a copy of the program on cassette or disc. All material should be original; if it is not chosen for publication, it will be returned to you.

You may not have written any software yourself, but you have very firm opinions about the world of Commodore and all their attendant industries and products. Then put your opinions on paper and post them to us, again at the address below — you never know, you might even get paid for airing your views! All submissions should be sent to:

The Editor
Your Commodore
Argus Specialist Publications Limited
No 1 Golden Square
London WIR 3AB

★PLEASE COMPLETE IN BLOCK CAPITALS

Your Name

Program Name

Computer/memory size it runs on

Amount of memory program occupies

Other computers/memory size which your program runs on without conversion or use

Does your game need or use joysticks? Yes No

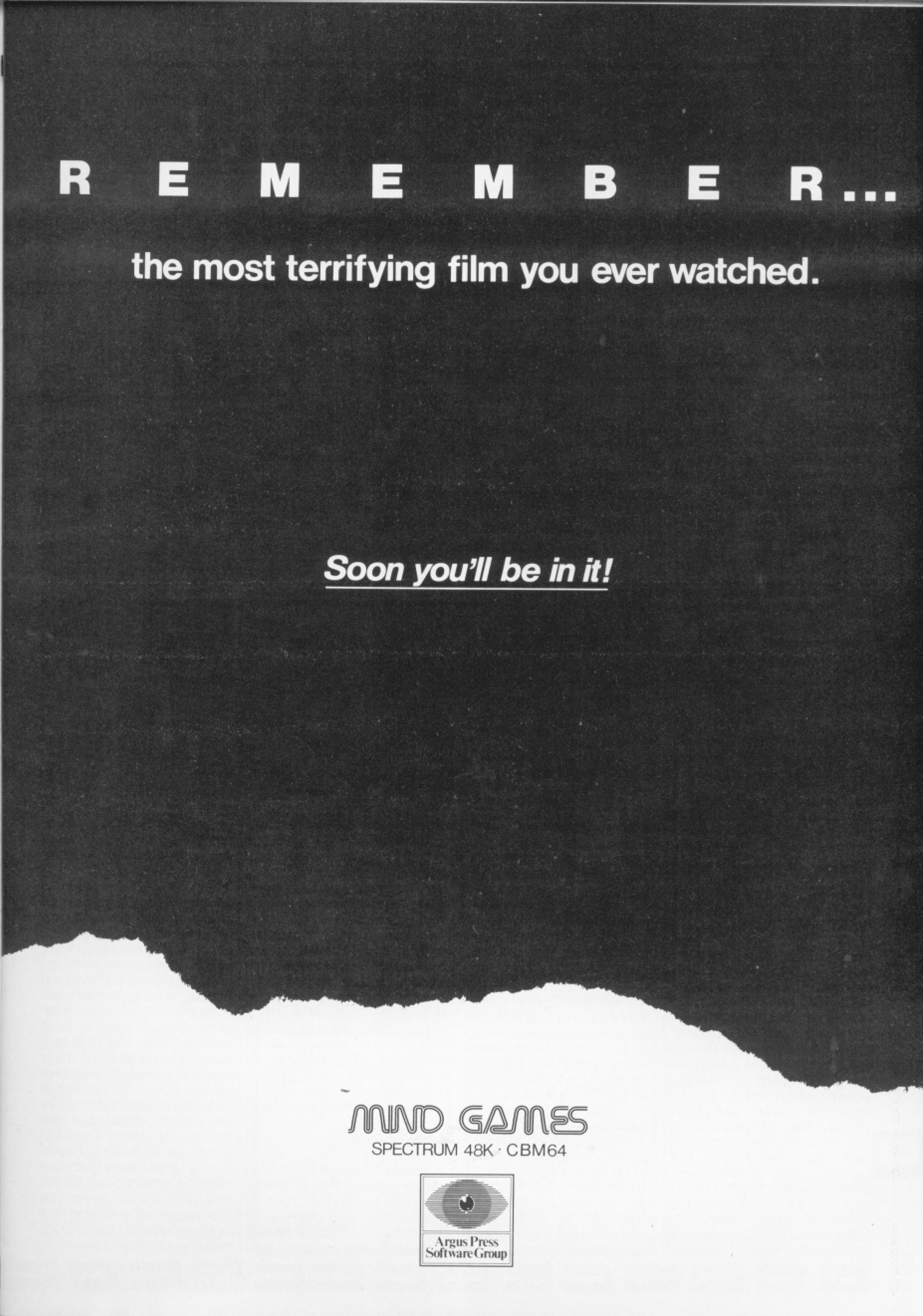
Have you sent our game to another magazine Yes No

Is it original/or a variation on a theme?

Your Address

Telephone Number

Times to contact you


REMEMBER...
the most terrifying film you ever watched.
Soon you'll be in it!
MIND GAMES
SPECTRUM 48K · CBM64
Argus Press Software Group

In pursuit of adventure, Pete Freebrey encounters magic, mystery and monsters in the second part of this series.

TALES FROM THE CRYPT



LAST MONTH WE LOOKED briefly at the original 'adventure' program created on mainframe computers and how, as personal computers improved, similar programs became available for the home computer. One of these was Level Nine's 'Colossal Adventure' which is closely linked to its original forebear. This is quite large enough — or should we say 'colossal' — to keep the ardent computer adventurer going for many days, weeks or even months! Although adventure games may come under many guises, the prime area still seems to come loosely under the banner of 'swords and sorcery', a generic title that has seen such a boom over the last decade in the world of the written word — principally in their paperback versions. This type of fantasy adventure, although not to everyone's taste, is very popular and certainly looks like providing us with plenty of varying plots for the forseeable future.

Some people are unable to relate to this form of 'other world': it is just too fantastic for them to grasp hold of its fundamental nature. No matter how well the program is written, either from the point of view of technical programming or from a literary standpoint, the basic structure does not turn them on. Without a wholehearted belief in what you are seeing on the screen, total involvement and, from this, total enjoyment cannot be achieved. Fortunately, the subject matter of adventure games is very diverse, ranging from the loosely historical, detective stories, space/science fiction and even popular TV series! Not all of these are yet available for the Commodore computers but, owing to Commodore's sale success, most should appear in the near future.

Defining an 'Adventure Game'

The term 'adventure games' covers a multitude of sins. The purist would probably say that they should be a cross between a crossword puzzle, a treasure hunt and a maze, the 'crossword puzzle' providing interlinked clues that enable the player both to find the treasure and, also, assist in mapping the maze. Paper and pencil are necessities to the serious games player since, in almost every adventure game, it is vital to be able to find your way around without either wasting moves (your lamp may go out too soon!) or being in danger of, for example, walking into a trap or being transported involuntarily to somewhere else! Mapping an adventure may be done in several different ways and in a future 'tale' we will look at this more closely; but, whatever you do, try to ensure that you know where you are and how to get back to where you were!

Early days

A looser interpretation of an adventure game is where you take on the role of adventurer and merely (?) have the freedom of choice in which way to move and subsequently find 'adventure'! Two early examples of this form would be 'Halls of Death' (Supersoft) and 'The Valley' (APS) — both of which are available for the Commodore 64 and the latter for the expanded VIC-20. In each of these programs you move around a mapped area shown on the screen; as you move you may encounter some form of unpleasant monster or an artifact that will increase your fighting abilities either magically or physically. Both have excellent real-time fighting routines that make the palms sweat and the heart beat faster. The CBM 64 version of 'Halls of Death' has a graphics representation of individual battles and 'The Valley' gives you the option of choosing one of five character types. Both require a certain amount of

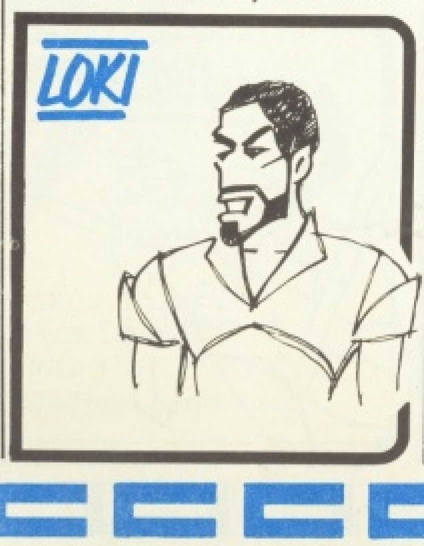
'tactical logic' on the part of the player to determine when to venture into a more difficult level. No puzzles are set and there are no mazes to solve but, as good old 'monster bashing' role playing games, they are still hard to beat. Both have character save facilities so your chosen character may progress in level and expertise over a period of weeks/months. Although both are now a little long in the tooth, they are well worth having on the shelf ready for an adrenalin boost when the nights are long and you want to vent your frustration on some poor unsuspecting monster!

Literary adventure

As we pointed out in our last 'tale', the pure text adventure offers the most scope for you and the computer programmer's imagination. On screen graphics take up quite a bit of your computer's valuable memory space (unless continually accessed from disc as will be many offerings in the future) and the graphics have to be good to make up for this limitation. Fortunately, every now and then there is a program that proves us wrong. The first to take up this challenge successfully was probably 'The Hobbit' (Melbourne House), originally available for the Sinclair Spectrum and subsequently converted to other machines, one being the Commodore 64. 'The Hobbit' broke new ground on several counts and must rate very highly in any 'adventure stakes'. 'The Hobbit' has a scenario based upon the classic book of the same name by J.R.R. Tolkein and a copy of the book (published by Unwins) comes with the computer program. Reading the book is beneficial, not only to get better acquainted with what the adventure is all about but also because it contains many hints on how you may play the game.

Over 50 scenes from 'The Hobbit' are represented graphically, obviously a great deal of care has gone into their production and finding new scenes is in itself a pleasure. The program has been written with the player in mind and several points have to be commended highly. The first concerns the graphics themselves: because these are on occasion quite complex, they may take a few seconds to draw. This is fine the first time you see them but could become boring and time-consuming if they were drawn on every entry to a particular scene. In 'The Hobbit' you will be shown a picture in all its glory on your first visit to that location, thereafter you only get the text description unless you specify LOOK. Some programs insist on performing a long-winded 'picture draw' on every entry and this can detract from the steady flow of the adventure. There are several examples of such increased playability within 'The Hobbit', one other being the use of ALL or EVERYTHING; it is so much simpler to type TAKE ALL rather than specify a list of individual items either as: TAKE ROPE (Return), TAKE SWORD (Return), TAKE FOOD (Return) TAKE KEY (Return), etc or even: TAKE ROPE AND SWORD AND FOOD AND KEY!







'The Hobbit' also provides a framework that is in itself interacting with your moves and commands. You will find Gandalf the Wizard and Thorin the Dwarf wandering in and out of your story apparantly of their own volition. They may even take various objects either lying around or in your (you play Bilbo the Hobbit) possession. Further interaction even allows for you to talk to or issue requests (commands) to these characters — SAY TO THORIN "CARRY ME" is a perfectly acceptable and actionable command.

Playing 'The Hobbit' can be a fascinating and rewarding experience. Melbourne House have even published a book called 'A Guide to Playing the Hobbit' that will help the frustrated to complete the game (well. . .perhaps!). Even with this book at one's elbow, the attraction of playing is hardly lessened — surely this could be said of a few games!

Whilst on the topic of playing guides or hint sheets, Level Nine have solved this problem in a very sensible and clever manner. Hint sheets may be obtained listing large numbers of, for example, 'things' — look up the 'things' — you are interested in (they are listed alphabetically) and you will find a number; look up this number in the answers and you will be given a useful clue. The answers are 'jumbled up' so that, if you really only want a clue rather than a big 'cheat', it is possible not to spoil your adventure by knowing too much! 'Colossal Adventure' for instance has some 320 'questions' and answers.

The Legend of Valhalla

Following the success of 'The Hobbit', Legend Software produced what









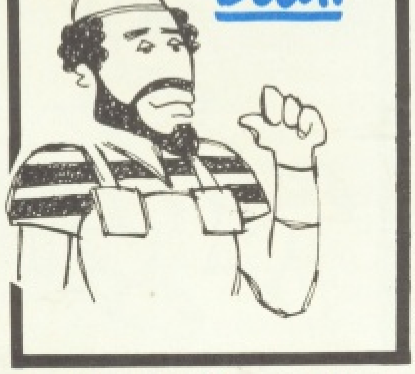
has become another classic adventure game — 'Valhalla'. This was very heavily promoted prior to its first appearance as a Spectrum program and appeared to offer an adventure game with animated graphics, with numerous characters who, it was said, could be converted to your cause, who would overhear your conversations and who may or may not be taking independent action on their own or someone else's behalf throughout the game. Once it became available, 'Valhalla' became almost an overnight success. Time passed and finally 'Valhalla' has now been released for the Commododre 64. 'Valhalla' may be viewed in several ways; it may be played as a quest, or looked on as a 'mini-movie' with you the player interacting as little or as much as you like to try and influence the outcome of the action. As a quest you must search for and obtain six specified items in a specified order.

'Valhalla' takes about eleven minutes to load and, for a good part of that time, you will have the title page to look at: this only shows the name of the company (Legend) and the game (Valhalla) but it's better than looking at a blank screen — perhaps you should be 'swatting up' on the instructions! Having loaded, you are given the option of loading a previously 'saved' game. Once into your adventure, you may save the game at any point — but you may only load in this data after the initial program load! So to move time back a little (to just before you lost something valuable) will take you about 14 minutes. Left to its own devices, 'Valhalla' will have various characters — gods, giants, dwarves, etc have different shapes to aid identification — appearing within the top two thirds of the screen: this is the graphics window within which you will see a picture of your location (always looking north). Various items — food, wine, rings, jewels, keys, etc may be visible and you are at liberty to pick them up, providing another character does not do so first! The lower third of the screen is split into a six line window telling you what is happening and a two line window where you may enter your commands. 'Valhalla' will accept simple one letter direction commands — N,S,E,W — and also move elaborate sentences such as SAGA PUT RING IN CHEST.

Time taken to draw each location is about 12-15 seconds and the time taken to action your commands varies enormously depending on what other actions (yours or those of the computer) are already on the 'stack'. This can be frustrating as you may wish to change a command because a certain character has entered the scene since you entered an, as yet, unactioned command! It can be a little difficult trying to type in what you want to do whilst the action continues on the screen. Creatures attack each other (and you) with monotonous regularity and whilst they are slogging it out 'on screen' you must patiently wait your turn. They really are an aggressive lot but, although many are killed (including you!) this is only a temporary setback and reincarnation seems to be the order of the day — everyone returning as strong as they were in the first place.

The instruction booklet provides a fairly clear picture of how you may do various things: eat, fight, move, buy, sell, etc but, understandably, does not tell you how to achieve very much! 'Valhalla' players seem to fall into two distinct categories — those that absolutely love it and those that think it is a total waste of time. There does seem to be a much greater leaning towards random action interferring with your idea of progress in most adventure games but, is this purely aggravated because you are not learning from your mistakes? I suspect that the 'story-line' is fairly thin but does demand that you walk a narrow (quick) path in the right directions. Deviate for too long and you will be lost in the random factors. Make maps and record what you do; do not get sidetracked too often! It is a pity that you cannot load a saved game at any time. If you foresake the quest, it is an interesting exercise to attempt the alteration of some of the characters' alignments — bad to good (or vice versa). 'Valhalla' is certainly a fascinating adventure and will surely prove to be a classic of its type. It is worth playing if only to find out your alignment — lover or hater! Write in and tell us which you are.









It's easy to complain about advertisements. But which ones?

Every week millions of advertisements appear in print, on posters or in the cinema. Most of them comply with the rules contained in the British Code of Advertising Practice.

But some of them break the rules and warrant your complaints.

If you're not sure about which ones they are, however, drop us a line and we'll send you an abridged copy of the Advertising Code. Then, if an advertisement bothers you, you'll be justified in bothering us.

The Advertising Standards Authority.
If an advertisement is wrong, we're here to put it right.

ASA Ltd, Dept 2 Brook House, Torrington Place, London WC1E 7HN

Braingames

The tough nuts to crack!



ELECTION TRAIL

For the Commodore 64

Disk £9.95
Cassette £7.95

BRAINGAMES Amplicon Group
Richmond Road Brighton East Sussex BN2 3RL
Tel: Brighton (0273) 608331 Telex: 877470 AMPCON G
Braingames is a division of Amplicon Micro Systems Limited

Braingames

The tough nuts to crack!



FAME QUEST

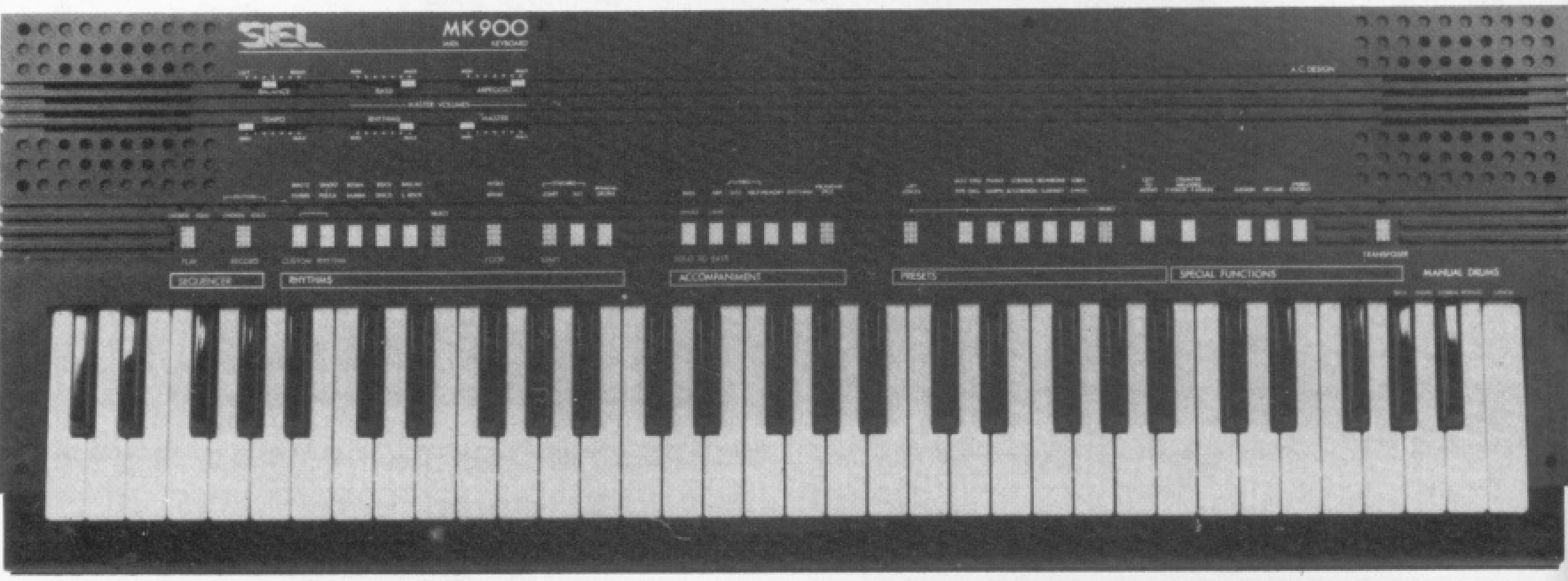
For the Commodore 64

Disk £9.95
Cassette £7.95

BRAINGAMES Amplicon Group
Richmond Road Brighton East Sussex BN2 3RL
Tel: Brighton (0273) 608331 Telex: 877470 AMPCON G
Braingames is a division of Amplicon Micro Systems Limited

Get in harmony with Chris Palmer as he dons his musician's hat and tunes into MIDI.

MIDI



REVIEWED

IT CAN HARDLY BE DENIED that for many people the home computer is a source of entertainment. For most this comes in the form of games, be they arcade, strategy or adventure.

A fact which is often overlooked when people buy their computer is that they are also buying one of the most creative tools mankind has ever built. The main reason for this is that, until recently, both hardware and software manufacturers have themselves overlooked this area, in favour of more 'arcadian' pursuits.

In this article I hope to bring to light one development in the computer field which has the potential for revolution equalled only by the Space Invader.

That is, the magic of MIDI.

From Beep to Bach

Back in the dim, mistenshrouded days B.S. (Before Sinclair), computers were created silent. In the back rooms of pubs and other secret meeting places, groups of users would perform strange rituals to give their computers the power to make noise. Generally, this would involve disembowelling transistor radios and soldering their vitals to the computer's user port (not their own, the radio's vitals!!)

Then, by chanting the dark and secret language of machine-code, they would toggle their outputs to produce a frequency. When heard through the radio speaker, this frequency would sound not totally dissimilar to a musical note.

This caused much celebration in the camp of users and pretty soon they were learning to change the frequency to produce tunes.

At last! The silence had been broken. Very soon the users were producing programs which would allow them to bore their lay friends silly with faltering single finger renditions of 'Ba Ba Black Sheep' and the like.

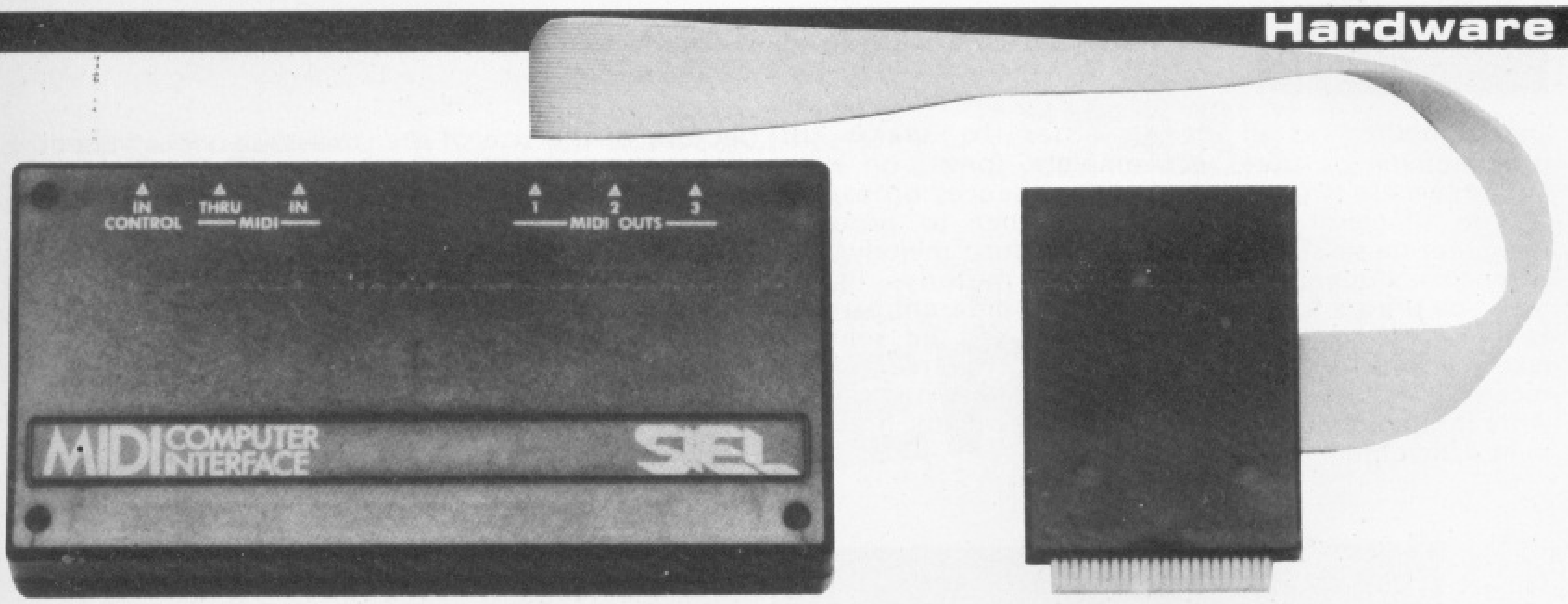
The manufacturers resented the users taking such liberties with their computers and when the threat of invalidated warranties failed to stamp out these arcane practices, the manufacturers decided to act.

Thus was born the musical computer. At first the computers were monophonic, meaning that only one note could be played at a time. The next was to give the computer multiple voices, thus making crude polyphonic (chords etc) playing possible. Often the computers weren't very accurate in their tuning and the quality of the voices left a lot to be desired.

Recently though, things have been looking a lot better. For instance, the Commodore 64 offers not only a selection of different wave shapes, but better tuning and the ability to set up sound envelopes. With these features it is now possible to make a computer sound quite like other instruments.

On the software side things have improved considerably. No longer is it necessary to code tunes the hard way, using numbers instead of notes and FOR. . NEXT loops instead of rests. Most modern day composer programs allow notes to be input in standard musical notation. Some even allow the user to use the typewriter keyboard as a sort of piano keyboard to input the information.

Here lies the problem, and one of the reasons for MIDI.



Whilst computers are the ultimate 'Jack-of-all-Trades', they will always be beaten when confronted by a device actually designed to perform the function.

Just what is MIDI anyway?

For the uninitiated, the phrase MIDI stands for Musical Instrument Digital Interface. As with any interface, it is a means of passing information between one location and another.

Taken in its simplest form, MIDI will enable you to plug one keyboard equipped with the interface into another and merrily send information backwards and forwards between them. A simple set up would be one keyboard 'listening' via MIDI to what is being played on the other keyboard. The information being sent from the first keyboard would be a number corresponding to the key being pressed on the keyboard. This would be sent in digital form through MIDI to the other keyboard. When this keyboard receives it, it 'plugs' it into its circuits so that the rest of the keyboard thinks one of its own keys has been pressed. All this happens very fast, so that what you end up with is two keyboards playing in unison.

Of course, this is only a small part of the information that can be sent via MIDI.

Anyone familiar with synthesisers will know that on most of them you can create a sound and store it in the synth's internal memory. This is known as a patch. During use, any of these patches can be recalled by pressing one or two buttons on the synthesiser, far better than trying to change the settings on 50 or 60 knobs and sliders.

A MIDI equipped synth will also allow you to send the patch number you are currently using through the interface. So if you are using two or more synths linked together you can change the patch number on one and also have the patches change on all the other synths.

All in all, what MIDI allows you to do in these situations is to duplicate what you are doing on one synth on up to 16 others.

How does it work?

The MIDI system is based around a serial data bus, similar in many ways to our old friend, the RS232. As with all things serial, it's got to happen fast. RS232 operations usually happen at around 19K, that's about 2K of information a second. The MIDI system can send and receive data at 31.25K. Not surprising when you think that it might be called upon to control sixteen keyboards at once.

Each data 'word' consists of ten bits. A start bit which is zero, eight data bits making up the information byte and finally a stop bit set at one. Anyone who has mucked about with defining characters on a Commodore will know all about turning an eight bit binary number into decimal, and vice versa, so I won't go into all of that now.

The format of the MIDI commands is more complex than just a single byte representing the note played. It must also take into account the information for the attack and decay rate of the note and which keyboard the note is to be played on. Also implemented is a system of control codes, which preface the information for patch changes, bends and other control information.

Enter the computer

So far we have looked at computers making music on their own and sythesisers talking to each other. Now comes the interesting part of putting them together.

What a computer is best at is manipulating data in one form or another. We also now have keyboards pushing data out of their MIDI sockets. The first task is to find a way to intercept this information and route it into the computer.

Luckily this isn't that much of a problem. Since the history of interfacing computers goes back a long way, it wasn't long after the advent of MIDI that the first computer interfaces became available. Because of the Commodore 64's popularity, a large number of these interfaces are for the '64.

Like all pieces of hardware, it is the software which really makes the system perform. But before getting too deeply involved with the soft side of things, it would be best to meditate on the nature of time.

Now before you get too worried I just want to introduce you to a new sort of time, step time. Everything that goes on around us is said to happen in real time. Now, if we could break all the events that happen into handy 'bite-size' chunks and have them happen to us when we wanted them to, time would appear to be made up of a series of steps, hence step time.

This is an important concept to hang on to when dealing with synthesisers

and computers as all the programming occurs in either of these two modes.

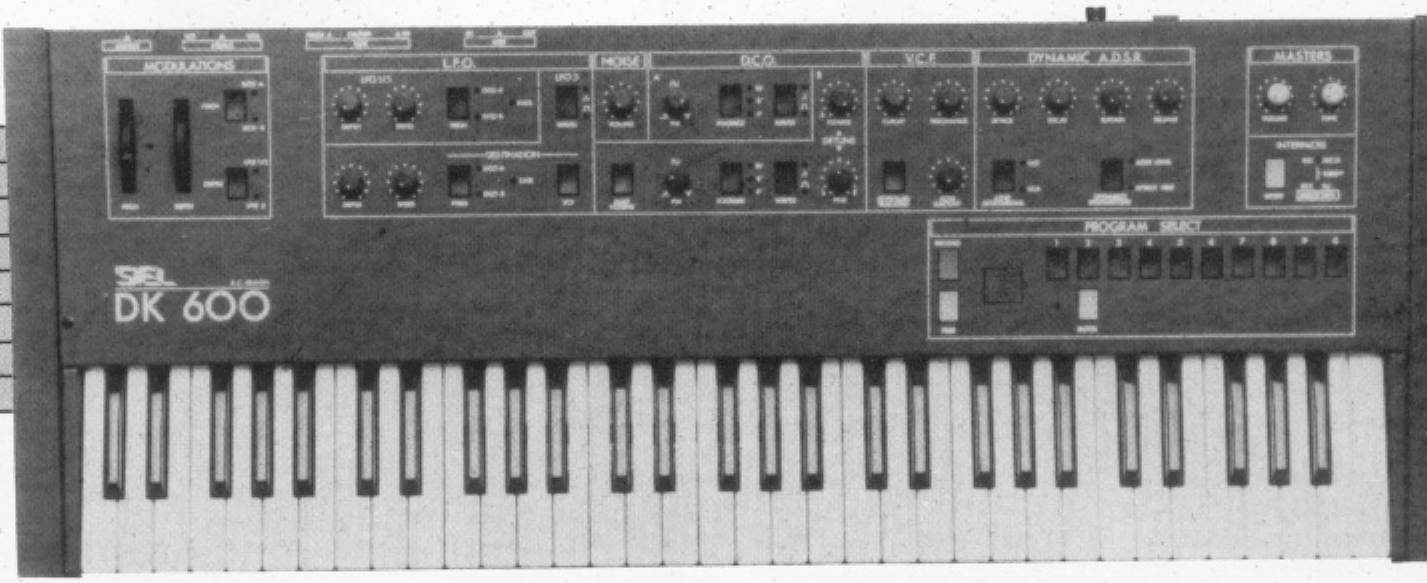
The simplest piece of computer music software is a step time sequencer. Every time you press a key on the keyboard the computer will take the information it has received through MIDI and write it into memory. It will then increment a note counter and sit there waiting for you to press another key. This process is then repeated until you tell the computer to stop recording.

To play back the piece all the computer has to do is reverse the procedure and place the information back out on the MIDI bus at the rate set by an internal metronome.

The disadvantage of this system is that it is very difficult to get any 'feel' to the music. It doesn't matter what dynamics you put into your playing on the keyboard; the computer will just trot the notes back out at you in strict time.

The big advantage of this way of doing things is that it is very easy to edit the piece once it is in the computer. If you have played a wrong note you can step forward through the recording one note at a time until you reach the mistake. If you then switch the computer into 'record' you delete the wrong note and play the right note in its place.

With the more advanced step time sequencers you can chain sequences together to make up complete songs, or even layer sequences on top of one another to produce bass, rhythm, melody and counter melody. Being MIDI, the different parts of the song can be sent to different keyboards and the patches on the song can be changed as many times as required.



For those who are more at home on the computer keyboard than one of those ones with funny black and white keys, there are some packages which allow you to input the note information in alpha-numeric form.

For those who prefer sitting down at the old 'joanna' then real time computer recording is for you. This software works in a similar way to the step time sequencer, except that it remembers every nuance of your playing style.

The way it works is to divide the computer's memory into a lot of very short events. When you set it going the computer runs through these events in real time. Anything which comes down the MIDI bus during this time gets put into the corresponding section of memory.

In playback, the computer plays the information back at the same rate (unless you tell it differently), therefore recreating exactly what you played into it. Once again you can chain passages together or overlay passages. Unfortunately because of the size of the note events, editing is virtually impossible.

To the future

At the moment, the whole field of computers as control devices is very much in its infancy. For the average home user, the thought of spending £500 plus on a synthesiser/interface system requires a lot of interest in the field of computer music. However, a couple of recent developments give some idea of the direction in which things are heading.

Firstly, a large number of the synthesiser and keyboard manufacturers are starting to produce stand-alone MIDI modules, primarily for use in conjunction with other synths. These devices have all the sound production capabilities of a full synth, but without the cost of elaborate controls and a keyboard. This makes them ideal for the computer based musical enthusiast who wants to explore music and sound, but who doesn't want to pay for a full keyboard orientated device, most of which he is not going to use. These modules are starting to become available for the cost of a few hundred pounds. Anyone who has observed the way prices of computer hardware have dropped can see that it is only a matter of time before they are available for less than the price of a disc drive or printer.

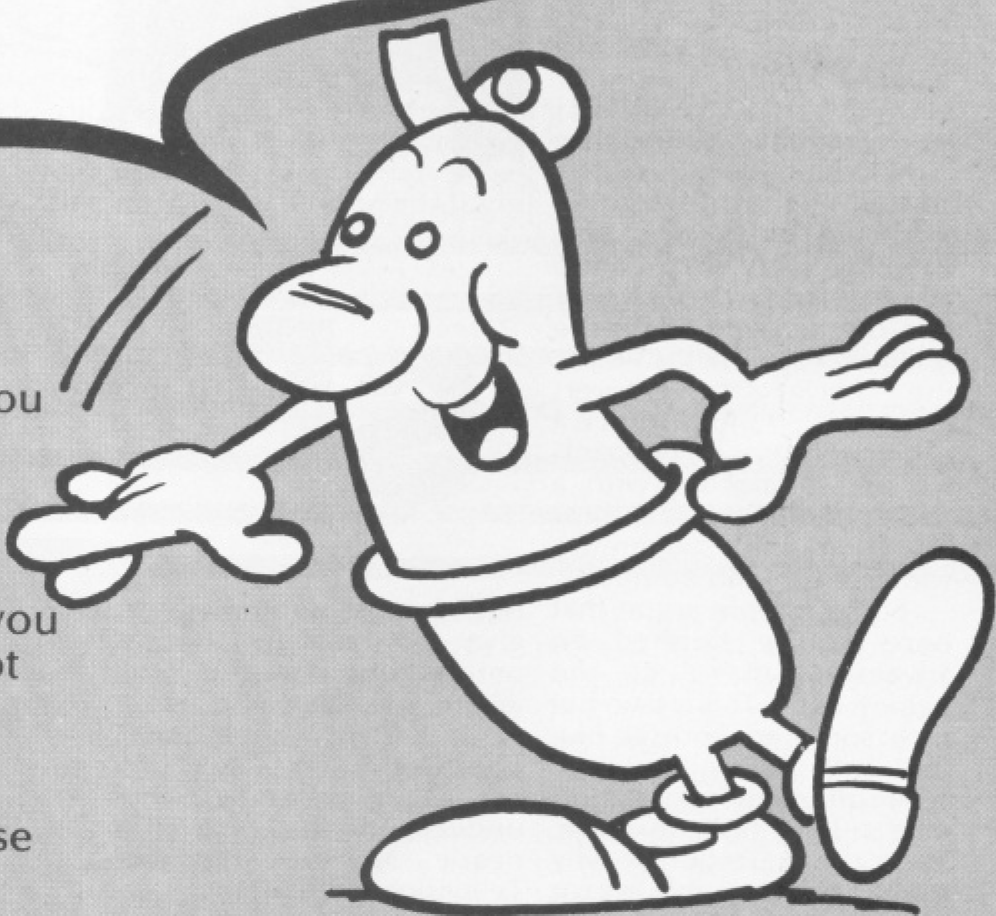
The other development comes from Casio. Already Casio have firmly planted themselves as the kings of the cheap home keyboard and have done much to make music accessible to everyone.

One of their latest keyboards has a Centronics interface built in and software is available to use this in conjunction with a computer. I would be very surprised if the next generation of Casio keyboards does not include some degree of MIDI compatability. With the reputation they have already got for producing quality home keyboards, they are in an ideal situation to bring affordable computer music into everyone's homes.

In the end

The computer has been a great equaliser in everything it has become involved with. The beauty of computer based music systems is that they can be extremely tolerant to whatever musical level you are at. You no longer need to be able to play an instrument to be able to express your musical thoughts. All there is stopping you now are the limits of your own imagination. But above all HAVE FUN!!!

CALL TO REVIEWERS!



You will probably have read the reviews of software in this issue of Your Commodore, of games and utilities and adventures. What did you think? Did you agree with what was said? Did you strongly disagree?

If you like using or playing with the latest software for your Commodore micro, and you think you can write clearly and wittily, why not see if your talents could earn you some money as well as fame?

If you would like to review software for Your Commodore please fill in the form below and send it, along with a sample review of your favourite program, to the Editor of Your Commodore at the following address:
Your Commodore,
Argus Specialist Publications Limited,
1 Golden Square,
London W1R 3AB.

If we think you've got what it takes to make our team of reviewers, we'll be in touch — you'll get to keep the software that you review and you'll get a small payment for your efforts too!

So don't delay — start writing now!

Name ....

Address ....

.... Post code. ....

Telephone number..

Commodore micros available ....

Peripherals available..

Particular area of interest .... Games. ....

Utility .... Business. ....

SOFTWARE SPOTLIGHT



Your Commodore's monthly overview of the software currently available for Commodore users.

THE DALLAS QUEST
★ ★ ★ ★ ★
U.S. Gold-Datasoft Inc.
£14.95
CBM64 + Disc Drive

THE PROBLEM WITH reviewing adventure games is that you have to play it in it's entirity (if you are going to be fair) and then not give anything away (or not too much at least). Now, if it is a good game it must have the following qualities to keep the player (adventurer) enthralled. It should, if it is a text adventure, be descriptive and humorous with a strong plot. The same goes for a graphics adventure with the obvious additions.

Some people argue that once you've finished an adventure that's it, the game's over! This is true but with some adventures like the Zork Series, they are very difficult indeed and take a long time to solve. The other question is why pay the prices people ask for adventures when normal games cost maybe half the price? If a lot of thought has gone into it and the planning is meticulous, then it should be worth the money.

The Dallas Quest is maybe one of the best adventure games graphics wise, but first of all let me tell you a little of the plot. You are a world famous detective who has been summoned to South Fork by Sue Ellen. The reason for this is that she would like you to recover a map that describes the whereabouts of a very rich oilwell so that Sue Ellen can become financially independent of J.R..

As far as Sue Ellen knows, the map is in the safe keeping of Jock Ewing's old friend "Chugalug Jones". Now this character is in South America running a trading post and will only give the map up to the person who has the ring which Sue Ellen also gives you along with the photograph of "Chugalug Jones". Right now, you understand the plot; it's nearly as good as the Television programme!

As with most new adventures on the CBM 64 that I have seen they all have some degree of humour built in. For example, on Dallas Quest there are two very obvious jokes. One is to do with the owl (no clues) and the other is when you get to a Cannibal Village, it says something about a feat of courage and you mistakenly hear him say feets of courage and start to dance. In this animation, three sprites of dancing girls are used.

This conveniently leads to the graphics and I must say that a great deal of thought has gone into it. Each movement leads to a new high resolution screen and I must confess that, as the game came close to it's conclusion, the screens got even better! The pictures of things like a giant statue of "Chugalug Jones" or the



'Airport' were very good indeed. The only disadvantage in having such high quality screens is they take a long time to be reproduced. Yet again, thought has been visibly used because the writer has put in the facility to switch off the screens so as to save time and enable you to get to the point where you got killed last time!

Now if you have played adventures before and wished you could save up to where you're about to try something dangerous, then this is the Rolls Royce of games in enabling that facility. In fact this program allows you to save 9 different games before you have to re-save or update a copy. When I played it, it was used to the full. Along with this feature you are given the chance to use 9 clues; you don't have to but you can if need be.

You will notice that throughout this review I have not mentioned music, the reason being that there isn't much use of S.I.D. chip at all and unfortunately what little there is does not score very highly with me.

So, finally having got through without dropping too many hints in this review, I must reach my conclusion. It is one of the best games out on the CBM 64; it shows that the 64's graphic capabilities are equal, if not better, than it's competitors and also shows that the software available on this machine is of a very high quality and gives credit to the programmers, along with Commodore.

And, if all else fails, there will be someone to listen to your strangled cries of anguish and maybe even help you!

S.L.F.P.

AZTEC CHALLENGE

★ ★ ★ ★

Cosmi — Audiogenic
£8.95 (tape) £12.95 (disc)
CBM64 + Joystick (Cassette and Disc Based)

ATTRACTIVELY PACKAGED, Aztec Challenge comes from the same author who wrote Forbidden Forest. Previously written for the Atari, the successful change from one machine to the other has been achieved. As with 'Forbidden Forest' a high standard of graphics and sound have been maintained throughout the game.

The game opens with the Cosmi logo which changes into an Aztec god's face. Then you have the option of either a one player or two player game. Once selected, the screen displays the scoreboard and then, after pressing the fire button, goes into screen 1.

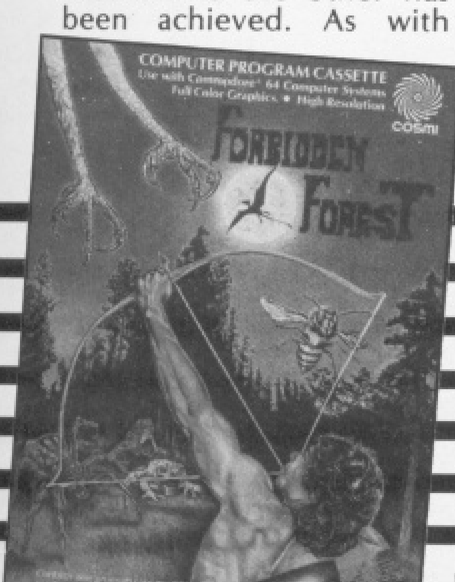
In the seven screens that follow youhaveto duck and jump spears on the way to the temple. Dodge the blocks of stone which roll down the temple steps on level 2. On level 3, when you've got into the temple,



you have to run through the various rooms, each of which have nasty traps to be negotiated. Once through that you meet the vermin that infest the temple; if you touch them for more than a second the venom which covers them will kill you.

Having escaped the creatures, you encounter a room with booby trapped tiles, a piranha infested lake and finally on level 7 the bridge which has some of the slats missing. Once you have completed these minor tasks (!), you return to the beginning and start again, only the game becomes a little harder. This is definitely one of the best games around and I recommend it to any budding athlete.

S.L.F.P.



FORBIDDEN FOREST

★ ★ ★ ★ ★

Cosmi — Audiogenic
£8.95(tape), £12.95 (disc)
CBM64 + Joystick (Cassette and Disc Based)

ANOTHER CLASSIC FROM Paul Norman, this has to be among my top five favourites, along with his Aztec Challenge. Although the graphics aren't totally first class (but very close to it) the game itself is amazing. The game story line is as follows. Apparently while walking one day you amble into what looks like a normal forest. This is a mistake because it is a forbidden forest.

In this 7 level game, you encounter more evil monsters than you would do visiting the Munsters' house! The game opens with you, bow and arrows in hand, ready to meet. . .the giant spiders. Once these monsters have been despatched you encounter more grisly creatures including Bees, Frogs, Dragons and then the Phantom.

It is worth pointing out on this level that the Phantom also has skeletons with it. The skeletons keep on coming until you shoot the Phantom in the hood and when you do hit him he disintegrates before your eyes. Now you meet the snake and finally our hero gets to grips with the Demogargon himself.

To make things a bit more difficult, while you've been fighting these lovable creatures, it has been getting darker! The Demogargon is a difficult creature to shoot. . .The only time you get a chance to take a pot shot is when the lightning strikes, lighting up the sky. If you don't hit him (which is likely) well. . .

A superb game with a good music piece and an addictive original idea.

S.L.F.P.

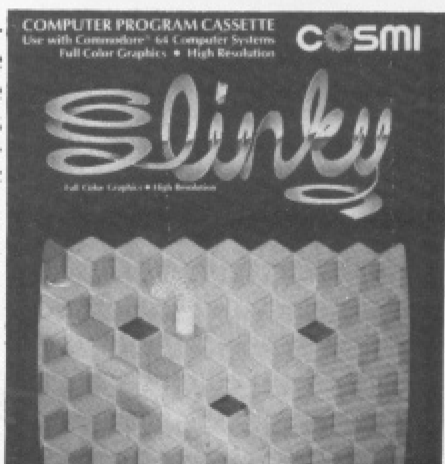
SLINKY

★ ★ ★ ★

Audiogenic — Cosmi
£8.95 (tape) £12.95 (disc)
CBM64 + Joystick (Cassette and disc Based)

THIS GAME COMES FROM the same people who produced high quality packages like Forbidden Forest and Aztec Challenge. Yes folks, they've done it again with Slinky. This game is a good quality reproduction of 'Q★BERT', but they have had the foresight to put a few enhancements into the game.

As usual with Audiogenic/Cosmi games the graphics are very good. The use of sprites in this game is to the full and, in parts, very clever. The animations of Slinky (a spring) jumping from one cube to another is superb! Anyway, the actual game involves jumping on all the cubes and changing the colour. To stop you are various characters, some of which can be useful at times and dangerous at others. For example, 'Ralph the Random Raindrop' can make you wet and you can then jump faster. However, if you are wet and 'Dusty the



Rust' touches you, you've had it!

There are various other characters in this game such as magnets and a metal head which appear from time to time. There are a couple of drawbacks with this game though: the scoring system which is a little too complex and, also, the fact that you can't select a level to start on.

There is one last addition that makes it worth playing: when you complete a screen without losing a life, a little character runs across the bottom and then shows you an action replay of how you completed it!

S.L.F.P.

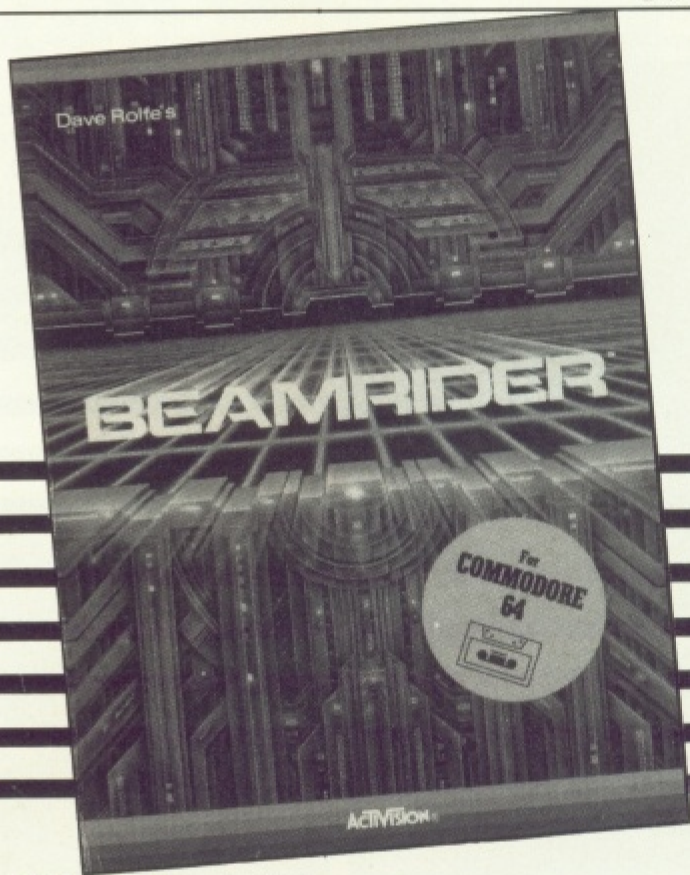
OXFORD PASCAL
★ ★ ★ ★
Oxford Computer Systems (Software) Ltd.
Approx £50.00 disc (unconfirmed at time of review): £15.00 Tape
CBM 64

THE GREAT PRICE DIFFERENCE between the tape and disc version appears to be due to the fact that the disc version can run compiled programs independent of the comp/ed program. This version claims to be a full implementation of Pascal and, from the time I have spent with it, this would seem to be true. This version also has extra commands to enable effective programming of sound and graphics. I enjoyed using them and it does indeed simplify things. The documentation is very good and there is even an errata sheet in the manual which corrects spelling mistakes. This would indicate that a lot of thought has gone into the preparation and presentation of the program and manual. Once one knows a language it is difficult to imagine how good a manual is at teaching a complete novice. I think that although it is clear and concise with good demonstrations a true beginner would need some extra books in order to fully benefit from what Pascal is capable of. Essentially, Pascal is intended for data handling and, like BASIC, it is a high level language. It would be wise to consider your reasons for wanting to learn Pascal as appx 50.00 is no mean amount to pay for software which may prove to be unnecessary. To sum up then if it is Pascal that you want then this would not be a bad buy.

D.A.C.

SOFTWARE SPOTLIGHT





KICK-OFF ★ ★ ★ ★
Bubblebus
£6.99
CBM 64 + 1 or 2 Joysticks (or keyboard) (Cassette Based)



NOW HOW CAN BUBBLEbus get it wrong sometimes and then come up with ones like this which are absolute winners? Never mind, but seriously now this is good. Bubblebus have taken the original table Football and put it on computer. The game that I refer to is the one with handles at the side which you twirl frantically trying to score goals against the opposition!

The way it is played is by using either one or two joysticks depending on whether you play a friend or the computer. I don't like the one player game because I always lose against the computer (same old story!), but two players and you've got the World Cup! The graphics are very good on this game as you can move your players from side to side and even watch them twirl as you kick the ball. The joystick operation is quite difficult at first but, once mastered, it can become quite fast.

The game is played over 8 balls and half-time is after the fourth ball, at which point you change ends. The ball speed can also be changed to suit expertise. It's worth the money and I hope that they will produce more old pub games in the future.

★ ★ ★ ★ S.L.F.P.

BEAMRIDER
★ ★ ★ ★
Activision
£9.99
CBM64

EVERY NOW AND AGAIN A really good, wholesome arcade zapping game comes along to completely restore your faith in programmers. Beamrider is just such a game. Operating in three dimensional perspective, the object of the game is to clear the restrictor shield that surrounds the earth by destroying the enemy saucers and sector sentinels. But at what stage you actually clear the shield is beyond me. The designer sends greetings from sector 26 but to what extent this is prowess or optimism I can't tell. Me, well I managed sector 14 with a fair degree of difficulty. The difficulty was in mastering the single beam movement of my gunship that the game so obviously requires. Points are awarded for all enemy craft shot down and each sector is cleared once 15 enemy saucers have been blasted. As every sector is cleared, the sector sentinel passes across the beams at the top of the screen but this can only be destroyed by using special bombs of which you have three. As it begins its approach it is immediately protected by special green blocker ships which home in on the beam you occupy. Each sector has several rejuvenators which, if caught, give you extra lives with which to fight the enemy.

K.M.

DECATHLON
★ ★ ★ ★
Activision
£9.99
CBM 64

YET ANOTHER ARCADE winner from Activision aimed at all those frustrated athletes with a hankering for the Daley Thompsons. Featuring all ten events of the real Decathlon - 100, 400 and 1500 metre races, long jump, high jump, pole vault, javelin, discus, shot put and 110 metre hurdles — this game gives you the opportunity to compete for the supreme accolade of the world's greatest compu-athlete. Although the game can be played against the computer, it is best played against a friend in order to introduce a true and necessary competitive element into the proceedings. Just as in the proper event, points are awarded depending on the distance thrown, the height jumped or the speed with which a race is run. The graphic representations of each event are nothing short of superb and coupled with a crowd that gives you a standing ovation when you break the 1000 point barrier, it all goes to make a game that is difficult not to enjoy although may seem a little too sedate at first. But be warned. It is extremely hard on the joystick. Running and approach speeds are achieved by furiously knocking the joystick from side to side. Even if your joystick doesn't give out at some stage, your wrist is bound to be struggling long before the final 1500 metre slog.

K.M.

PESKY PAINTER
★ ★ ★ ★
Super Soft
£6.95
CBM64 + Joystick (optional)
(Cassette Based)

THIS IS A VERY GOOD version of a very good arcade game. Pesky Painter is a new version of a game called Amidar. Originally an arcade machine game when produced for the Atari VCS machine, finally it has come to the Commodore.

For those who aren't familiar with the game, a description is in order. Peter the Painter has to clean the spots of dirt off the palace walls but Peter is, unfortunately, lazy. The king of the palace tells his servants to keep an eye on him and if Peter stops for a brief moment he gets thrown out.

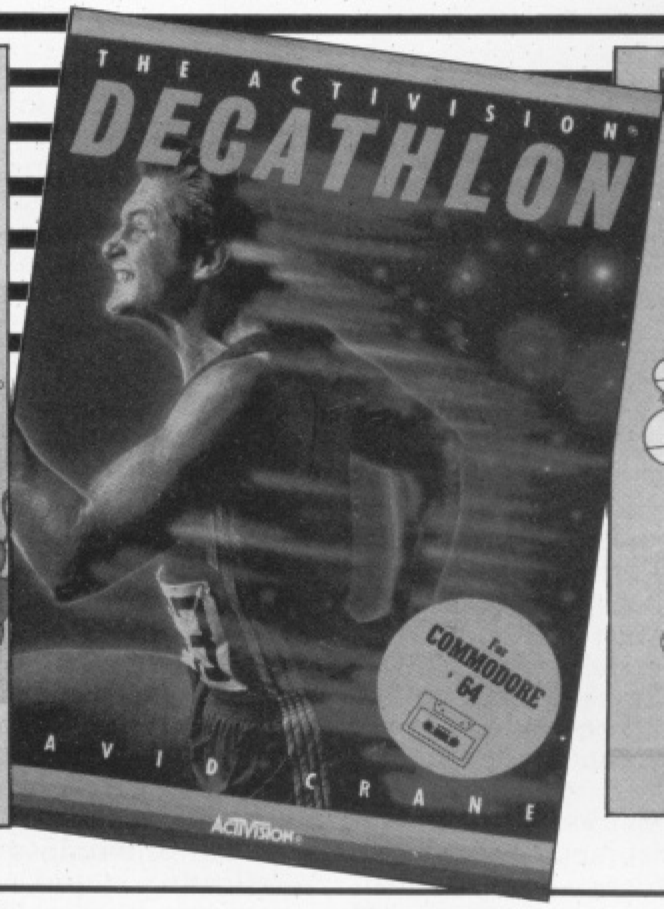
Once the Palace is clean, Peter has to feed the king's pet. To accomplish this, you have to choose the right route through the maze (I won't say how!) and when the pet is released it will go the route you have chosen.

The next screen is similar to the first, except that you now have to paint the walls. The way Peter does this is to run round the outside of each square. When the square is completely cut off from the next by paint it fills in and you get the points inside that square. There is also an incentive, in the form of the following: if you fill in the four corner squares you get the chance to catch the guards carrying more paint.

Good graphics have been used on this game along with a single catchy base line, which is now in my head!

S.L.F.P.







FRANTIC FREDDIE
★ ★ ★ ★
Audiogenic
£12.95
CBM 64 + Joystick
Based)

WHEN THE PROGRAM IS first loaded the screen displays the top ten high scores table along with the credits, and credit is definitely due to the two gentlemen who designed this game.

It says on the package 'Frantic Freddie' — "A game with a sense of humour", and I would agree with that statement. The game has an ongoing music track which has some old rock classics, such as Queen's 'A crazy little thing called love' and E.L.O's 'Don't bring me down'; it also includes a little ol' ragtime as well.

The game itself seems simple enough to start with but it isn't so: Frantic Freddie has to collect all the gold on the screen but he has to avoid the Greeblies who own the gold. To accomplish this task Freddie has to run up and down telegraph poles (he works as a telephone engineer) on the various levels of the screen.

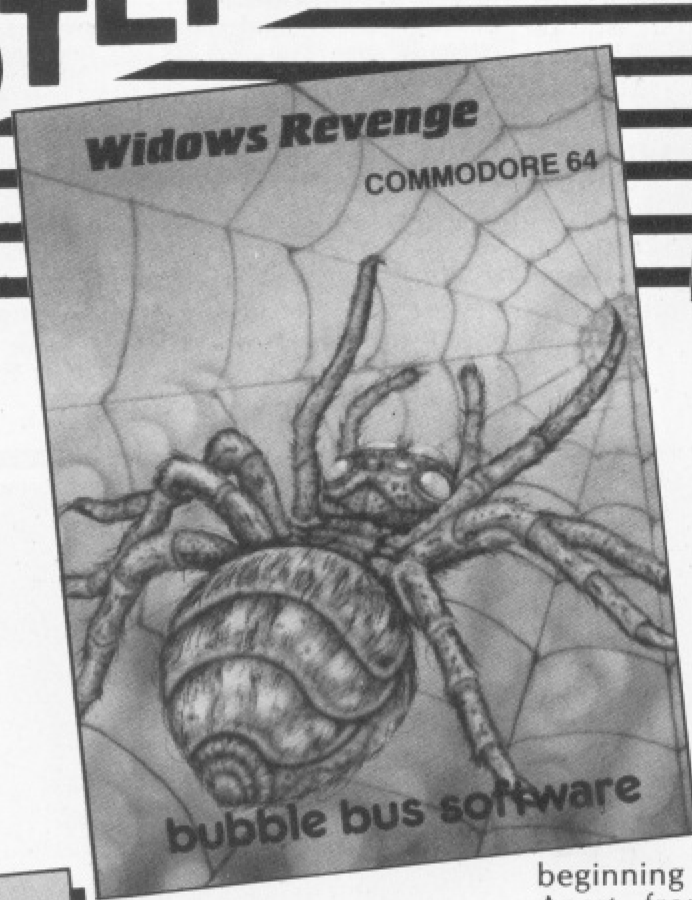
When two screens have been completed, a brief interlude of a cartoon sequence with the aforementioned E.L.O. music is completed leading you into the next two screens. I won't describe what happens in these cartoons but if you get a chance to see it, it's well worth it.

A stunning graphics and music game worth playing.

S.L.F.P.

SOFTWARE SPOTLIGHT





WESTMINSTER
★ ★ ★
Mr Chip Software
£5.50
CBM64

COULD THERE BE A Liberal revival? Might Kinnock slip on another seaside banana skin? And could Maggie be forced to eat her pearls of wisdom?

As a game of strategy Westminster has it all, even an independent party to occasionally upset the apple cart. Accommodating up to four players each of whom takes charge of one of the political parties, Westminster has the feel of a computer board game about it. The object of the game is simply to win the General Election which is achieved by canvassing the 60 constituencies and spending the campaign funds allotted to you by Central Party Office as wisely as possible on the campaign trail. Battle commences with £8000 stashed away in your coffers for which every £100 spent in a constituency can be expected to gain you between 95 and 105 votes. A personal appearance in a constituency will automatically gain you between 475 and 525. Seats are only considered to be safe once you have a lead of at least 1000 votes over your rivals. Additional funds from Central Party Office are allotted at every half and full constituency circuit completed. All campaign funds should be used wisely with each player ideally broke at the end of the number of canvassing rounds chosen at the beginning of the game. Apart from the 60 main constituencies there are 15 random outcome boxes which can gain or lose you funds and votes or enable you to visit 3 marginal constituencies or any of the 60. Every ten rounds of canvassing an opinion poll forecasts the result of the election. However, come the day of reckoning each party can gain or lose up to 500 floating voters which can throw the polls out completely if a lot of the seats are marginal. Although almost entirely a textual game with the minimum of graphical representations, Westminster is an enjoyable game of strategy which brings out the baser elements of political rivalry requiring a tactical understanding that is relatively simple to master.

K.M.

WIDOWS REVENGE
★ ★
Bubblebus
£6.99
CBM64 + Joystick (Cassette Based)

THIS IS, YET AGAIN, another Centipede game and it is from the same company who released Exterminator for the 64, which is again a Centipede Clone. Widows Revenge is, however, different in some aspects in that the Centipede is now a large amount of Spiders and these Spiders shoot back!

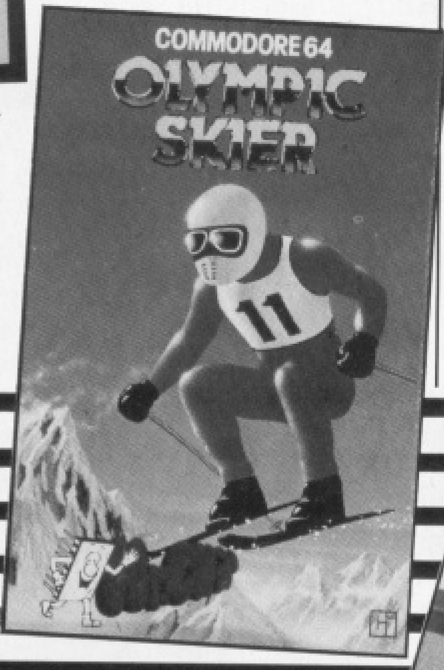
In all fairness to the programmer, the game is very good. It does have an addictive quality and the graphics are of a respectable level, but I really do think that Bubblebus should have released either Exterminator or Widows Revenge, not both.

The game itself is about a bird which lays eggs. Now, if one of these roaming spiders hits the eggs it stops and shoots at you. The main object of the game, therefore, is to eliminate the spiders and also shoot the bird (which will return to the screen after a short period of time). To gain points in this game you have to shoot the eggs and shoot the spiders, birds and anything else that comes along.

S.L.F.P.

OLYMPIC SKIER

★ ★ ★
Mr Chip Software
£5.99
CBM64



SO, YOU HAVE ASPIRATIONS of becoming an Olympic Skier. Well this game has all the necessary elements — slalom, downhill and ski-jump — yet, unfortunately, remains fairly uninspiring. Your objective is to achieve a maximum score of 1000 points spread over the three events. The first event which carries a four hundred point maximum score is the slalom. Here you have to ski the course paralleling left and right to take in the gates. For each gate taken in points are scored and there is a margin for error of three gates. Miss three and you are immediately disqualified. After the slalom comes the ski-jump. Accelerating down the ramp you must take off at the end and land on you skis without tumbling over in order not to have your score penalised. You are in complete control of the skiers acceleration, take off and landing. The downhill carries the biggest maximum score of 500 points. Here you have to ski down a special course avoiding the trees and jumping over all obstacles that get in your way. At the end of it all and, doubtless without completeing a course properly, you will emerge with a pretty lousy score with the game programmed to add insult to injury.

K.M.

MACHINE CODE TUTOR

★ ★ ★
New Generation Software
£14.95
CBM64

TWO TAPES AND A manual make up this package, with a different program on each side. The manual is not a nice thing at all; it looks as if it has been thrown together on a 40 column printer and pushed between a glossy cover. The information given in the manual is scant but what is there is accurate. I feel another book would be required by the beginner in order to fully explain what is going on. A review is a very personal thing and something that is enjoyed by one person may be hated by another. To me, the programs were excellent. I enjoyed the very original teaching method and the ability to step through a list of mnemonics with an explanation of what each was doing was really very good. It is easy to go back to little bits that you did not quite understand and just as easy to go forward if you come across things you already know. This is the first machine code tutor that I have come across that really uses the machine as a teaching aid. My only criticism with the program is that at times the choice of colours makes some parts of the program hard to read but that is all. It will not teach you machine code in just a few hours but with perseverance it should prove an invaluable aid in teaching some of the finer points of writing in machine code.

D.A.C.

HUSTLER

★ ★ ★
Bubblebus
£6.99
CBM64 + Joystick (optional)
(Cassette Based)



THIS IS A GOOD GAME for pool enthusiasts or for people who just want to potter about on the pool table. In this particular package you get six games for the price of one. Games which include three one player games and three two player games.

I must confess that I am not a very good pool player and can never get the balls in the pockets (except the white!), so when I started to play it was with reserved feelings. My feelings were magnified further with the presentation and the graphics on this game. But the actual game content made up for this.

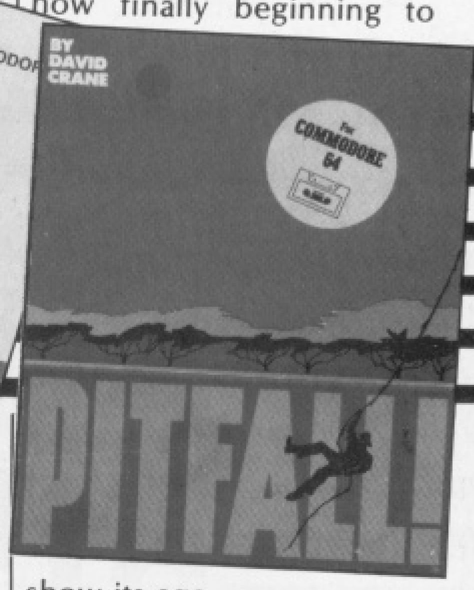
Obviously a lot of thought has gone into the various games contained in this program with selections such as — put any ball in any pocket being easy to cope with or at the other end of the scale — put each ball in its own pocket. To make things easier, at the bottom of the screen there is a potting strength meter and the cue is represented by a cross which you line up with the shot you have in mind.

S.L.F.P.

PITFALL

★ ★ ★
Activision
£9.99
CBM64

COULD THIS BE THE OLD Atari game converted for the 64, you may ask? The answer is yes and it is a little surprising how sedentary it now appears. Perhaps it is now finally beginning to show its age.



The object of the game is to guide Pitfall Harry through the jungle to find and collect 32 pieces of treasure including diamond rings, money bags and gold and silver bars. All this has to be done within a 20 minute time limit. Harry actually starts the game with 2000 points tucked under his belt. Every time he falls down one of the holes hidden in the jungle he loses 100 points and, similarly, every time he gets run over by one of the marauding logs. He also starts the game with 3 lives but there are several ways that he can lose them as well. Scorpions and cobras have to be avoided at all costs as do the crocodiles, although Harry is agile enough to jump on their heads when their mouths are shut. There are also the swamps, tar pits and quicksand to avoid although handily placed swinging vines can provide the necessary escape route. To find the treasure, Harry has to use the underground passages as well as the jungle...but rather you than me mate because I just found the whole thing a bit too damn silly.

K.M.



PEGASIS
★ ★ ★
Audiogenic
£12.95
CBM64 + Joystick (Disc Based)

IN THIS GAME FROM Audiogenic we are taken back in time to the days of myths and magic. The particular myth we are concerned about is that of Pegasis the winged horse.

As always in these trouble times there are the good guys and the bad guys. You are, of course, the good guy in white and the bad guys are in black. The idea of this game is to knock the bad Pegasis riders from their mounts without being knocked off yourself. To make life a little more difficult, if you hit the riders from underneath you fall off. The other drawback is that you will usually be outnumbered 2 to 1 or, as you progress through the levels, 3 to 1 or 4 to 1!

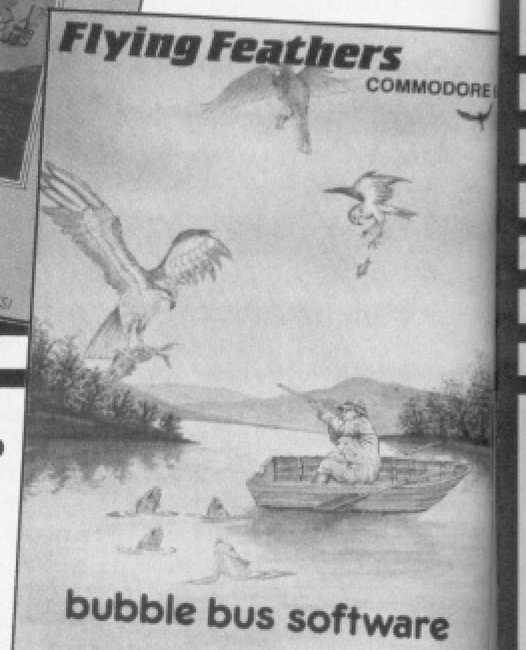
This game can be played with one or two players so team work or strategy should be planned carefully. Back to the game and, as the levels increase, there will be extra hazards to overcome. For example, on level 5 you meet a dragon and on level 9 you meet a buzzard.

The game is quite good with excellent graphics. The sprites being used are very detailed and the wings of the horse flap with the movement of the joystick.

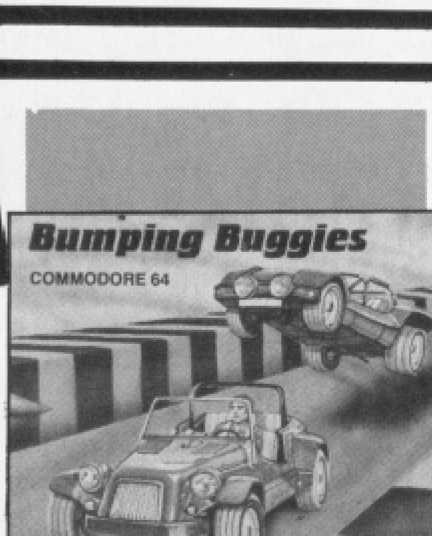
S.L.F.P.







BUMPING BUGGIES
★ ★
Bubblebus
£6.99
CBM64 + Joystick (Cassette Based)



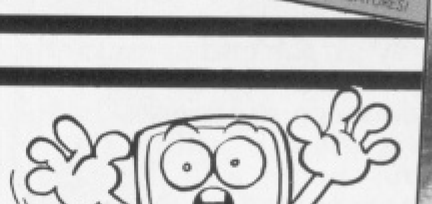
THIS IS A RACE GAME with a difference and the difference is that it is difficult. The idea of the game is to get as far as possible in the twenty levels while collecting as many points as possible.

The collection of points can be done in many ways. Firstly, just by driving you accumulate points as long as you don't crash. Then the other ways are as follows: you collect points by bumping or crashing your fellow drivers' cars or by only bumping them on the particular level you are on.

The level you are on also dictates which obstacles you have to overcome, from relatively simple levels at the beginning to totally insane levels later on in the game. Some obstacles I encountered, like the road stopping and having to jump from one island to another to reach the other side, proved too much for my fingers and joystick!

The graphics aren't the best in the world but it is still quite addictive.

S.L.F.P.



FLYING FEATHERS
★
Bubblebus
£6.99
CBM 64 + Joystick (optional) (Cassette Based)

IN RELEASING 'FLYING Feathers' I think someone has dropped an egg! It isn't the best game I have seen and even though it is an original or relatively original ideas, it lacks the all important 'pzaz', that extra something that makes it a game worth remembering.

The idea of 'Flying Feathers' is to stop the marauding eagles from taking all your fish. You being the gamekeeper means it's your job to shoot the eagles (no wonder they are endangered!) and save the fish. Occasionally a duck will quack its way across the bottom of the screen. This signifies that you have been awarded an extra fish due to your increasing score.

It appears that a lot more could and should have been done with this program as the graphics aren't exactly superb and it is a very slow starting game. I must confess that I felt more sorry for the poor eagles than I did for the blood thirsty gamekeeper.

This game has eight levels of play with levels 4-8 allowing you to shoot as far or as close as you wish. This offering from Bubblebus is definitely not for the animal liberationists!

S.L.F.P.

SYSTEM 15000

★ ★ ★ ★
Craig Communications
£9.95
CBM64 (Cassette or Disc)

THERE HAS BEEN SOME misconception in the general software market that the above title was a utility. Far from it, it is in fact a game. The basis behind the game is that a friend's company has been ripped off by Reako, another large company which has been infiltrated by organised crime. It's your job, by using System 15000, to retrieve $1,500,000 dollars and return it to your friend's bank account.

How do you do this? Well, with the aid of this System 15000 which is a telephone modem package (not a real one!) you phone various computers and gain access to special data. The real fun is when you find that the only help you have to start with is a telephone number of a Polytechnic and the password for a Company.

Two points to remember about this game is that it is very original and it's very frustrating! There are a couple of disadvantages, the first being that there aren't enough instructions, after all not everyone knows, how to operate a modem. The other is, whilst playing this game, there are certain times, when the police are checking the modem link and you have to go offline for a long time, a little excessive perhaps!

I recommend you play this game if you can. It could become a cult game in years to come and listen to the dialing tones as they are relevent to whichever country you are calling at that time (talk about detail!)

S.L.F.P.

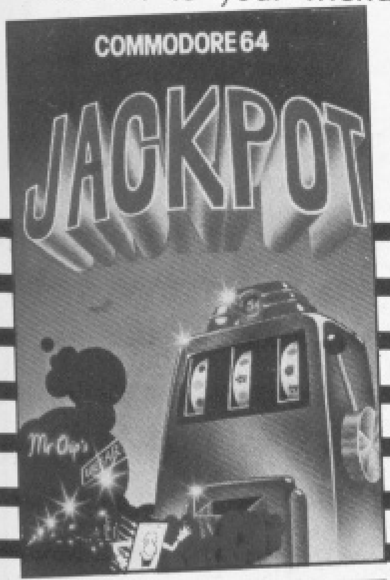
BOZO'S NIGHT OUT

★ ★
Target Software
CBM64



IT IS YOUR TASK TO SEE Bozo safely home from the boozer where he has been swilling pints and pints of wobble juice providing, that is, that you want to. There are two ways you can guide him home — the long way and via the short cut. The long way home is also the obvious way, along the road and Bozo's home can be reached either by turning left or right out of the boozer. The road is full of temptingly pretty girls desperate to stop Bozo from making it to his destination and other obstacles such as marauding grannies and arrest-happy policemen. Bumping into any of these people can result in the loss of his reserve pints of wobble juice. Losing all five spare pints will lose Bozo the game. Alternatively, there is the short cut through Weirdo Park where there are some far more unpleasant obstacles to avoid. Hiding behind the vegetation has its advantages here. Once home, you can either end the game or simply start all over again with Bozo tanked up with more wobble juice with the object of recording the highest score in the infamous League of Inebriates. Bozo's Night Out, I have to admit, may not be an entirely captivating game as it tends to be repetitive almost to the point of being somewhat boring but it does have some very good graphics which the makers claim to be in 3D. But then graphics alone do not maketh a game, so on this front Bozo does not rate much more than the joker he is meant to be.

K.M.



JACKPOT

★
Mr Chip Software
£5.50
CBM 64

TO REALLY ENJOY THIS game you have to be a complete fruit machine fanatic and, quite frankly, I can't believe that anyone who is that kind of fanatic is likely to be messing around with computers. In short, Jackpot is a dodo of a computer game. But still, if cherries, lemons and plums be the food of addiction then play on. To win the game you have to turn your £100 stake money into £250. Each spin of the four wheels costs a mere £1. Wins can be achieved two ways, either by lining up successful fruit lines or by the numerical value of the win line exceeding six in which case you are entitled to one or more shuffle wins. A hold facility enables winning lines to be more easily achieved and a gamble feature enables winning lines to collect anything between £3 and £100 depending on your nerve. In a nutshell, that is it. The verdict. . .well it has to be purely for the fruit and nutcases among you.

K.M.

EXTERMINATOR

★ ★
Bubblebus
£6.99
CBM64 + Joystick
(Optional)(Cassette Based)

THERE IS AN EXCITING cover on the front of this software package depicting a space age man shooting a giant worm! But when you put two and two together you've got an old idea in a new package.

It must be said that the version of Centipede which I have on my computer by Bubblebus is quite well written. It contains extensive use of graphics in the form of sprites with the inclusion of such creepy-crawlies as mosquitos, scorpions and tarantulas and even an eagle (not as much a creepy-crawly but still a hazard just the same). The sound was also extensively used, maybe a little too much but the programmer has added the facility to switch off the effects.

Also included is a pause button, just in case the phone rings while you are playing for that most important high score. The high score is displayed at the top of the screen throughout the game. If you pass the high score when you finish the game, you can type your name in and fame is yours at last.

Even though it is a copy of Centipede and that game is well known there seem to be the barest essential instructions. But every good game has to have an Achilles Heal!

S.L.F.P.

Get the thrills of the race track in your own living room with Simon Fong's great Grand Prix program.

FORMULA ONE

THIS GAME IS A GRAND Prix simulation with a different twist. Being totally confident of your driving ability, you have anticipated your win in advance and celebrated before the race! As a result, you are drunk (with alcohol, not success!) and you end up driving the wrong race in the wrong direction!

You have to try to dodge other racing cars coming towards you and also avoid oil slicks on the track. At the same time (if your co-ordination can stand it) you must try not to crash your car into the side of the road. Your task is to complete fifteen laps unscathed. The controls are: Z — left, M — right.

Don't forget, you are only allowed three crashes, so get out there behind the wheel and drive the race of your life!

Line explanation

INITIALIZE SPRITE	0 - 9
PRINT LINE OF TRACK ETC	10 - 54
GET KEY PRESSED	100
MOVE CARS	110 - 130
SOMETHING IN ROAD?	140
CRASH ELSE NEXT LINE	100 - 240
CRASH SPIN OUT	500 - 580
THREE CRASHES?	1000
END,ELSE START AGAIN	1000 - 1050
DATA	9000 - 9120

Variables

SP = SPEED
LA = NO. OF LAPS
OS = OBSTACLE (OTHER)
C = OBSTACLE (CAR)
SK = SKILL LEVEL
P = POSITION OF OBSTACLE
L = NO. OF CRASHES
X = YOUR POSITION

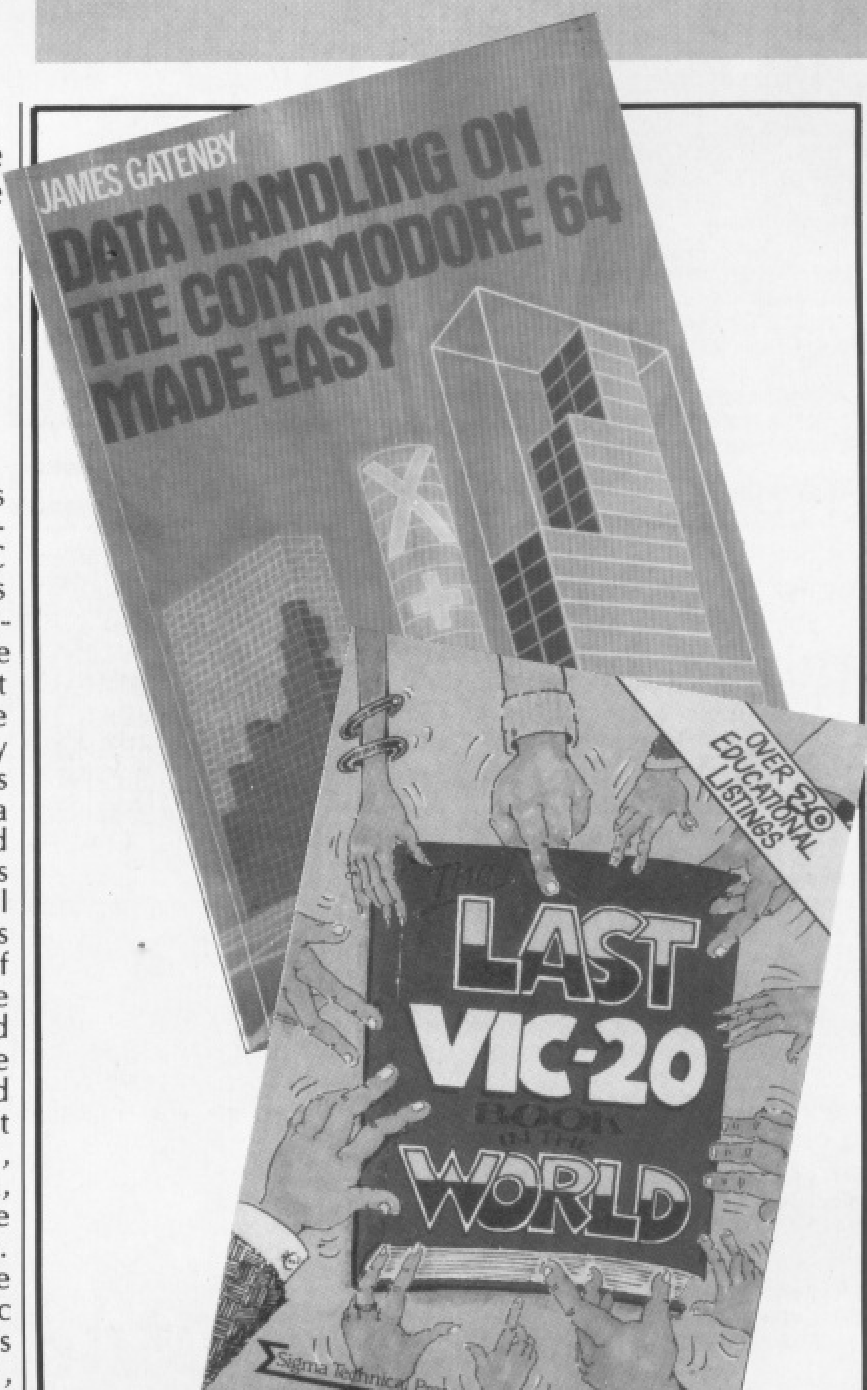


Program Listing

```
1 W1=54276:H1=54273:L1=54272:A1=54277:POKE54296,10
2 W2=54283:H2=54280:L2=54279:A2=54284
3 W3=54290:H3=54287:L3=54286:A3=54291
4 REM **** FORMULA **** BY WILLIAM &          SIMON FONG
5 PRINT"■■          WILLIAM & SIMON FONG":POKE53281,0:POKE53280,0:X=140
6 PRINT"■■■■■■■■■■■"TAB(14)"FORMULA ONE"
7 PRINT"■■■■■■"TAB(9)"■■ JOKSTICK IN PORT TWO ■"
8 AA=PEEK(56320)AND16:IFAA=0THEN14
9 Q=INT(RND(1)*3)+10:POKEW1,0:POKEW1,33:POKEH1,Q:POKEL1,Q+10:POKEA1,12
10 POKEW2,0:POKEW2,33:POKEH2,Q:POKEL2,Q+10:POKEA2,12
11 POKEW3,0:POKEW3,33:POKEH3,Q/2:POKEL3,Q/2+10:POKEA3,12
12 FORM=1TO250:NEXT:GOTO8
14 DATA0,0,6,6,6,6,0,0,255,60,60,126,126,126,255,255
15 DATA0,0,96,96,96,96,0,0,1,3,7,7,7,7,7,7,255,255,195,189,255,195,255,255
16 DATA128,192,224,224,224,224,224,224,0,28,28,28,28,28,28,0
17 DATA255,126,126,126,126,226,126,255,0,56,56,56,56,56,56,0
18 DATA126,126,126,126,60,60,60,60,3,15,14,14,58,0,0,192,192,192,192,252,172
19 DATA14,14,58,58,236,243,3,0,171,188,240,192,176,176,176,192
20 POKE55,255:POKE56,48:V=53248:SP=0
21 POKE53281,5:POKE53280,5:POKEV+22,PEEK(V+22)OR16:PRINT"■"
22 POKE53265,151:FORAD=0TO108:READA:POKE12*1024+AD,A:NEXT:POKEV+24,28
23 LA=0:X=150:POKEV+34,4:POKEV+35,7
24 FORAD=0TO62:READA:POKE832+AD,A:NEXT:POKE2040,13:POKEV+39,7:POKEV+37,0
25 POKEV+38,2:POKEV+28,1:POKEV,X:POKEV+1,100:POKEV+21,1
26 FORAD=12544TO12551:POKEAD,0:NEXT
27 PRINT"■";:FORN=1TO25:PRINT"■■■■■■■■■■■I■■■■■■■■■■■■■■I":NEXT:A=PEEK(V+31)
28 LA=LA+1:PRINT"■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■I■■■■■■■■■■■■■■I"
29 PRINT"■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■";
30 POKEW1,0:POKEW1,129:POKEH1,3:POKEL1,103:POKEA1,190
31 POKEW2,0:POKEW2,33:POKEH2,3:POKEL2,8:POKEA2,190
32 POKEW3,0:POKEW3,33:POKEH3,8:POKEL2,23:POKEA3,12
33 IFOS>0THEN38
34 IFCR>0THEN42
35 IFINT(RND(1)*10+1)<7+SKTHEN47
36 IFRND(1)<.95THEN41
37 OS=1:P=INT(RND(1)*11+1)
38 IFOS=1THENPRINTSPC(P);"■JK";:OS=2:GOTO40
39 IFOS=2THENPRINTSPC(P);"■LM";:OS=0
40 IFCR=0THEN47
41 CR=1:P=INT(RND(1)*10+1)
42 IFCR=1THENPRINTSPC(P);"■@■A■B";:CR=2:GOTO47
43 IFCR=2THENPRINTSPC(P);"■CDE";:CR=3:GOTO47
44 IFCR=3THENPRINTSPC(P);"■F■G■H";:CR=4:GOTO47
45 IFCR=4THENPRINTSPC(P);"    ";:CR=5:GOTO47
46 IFCR=5THENPRINTSPC(P);"    ";:CR=0
47 AA=NOTPEEK(56320)AND15:IF(AAAND4)THENX=X-4
49 IF(AAAND8)THENX=X+4
51 POKEV,X:POKEV+1,100:IFPEEK(V+31)=1THEN59
52 IFLA<300-(SK*2)THENGOTO28
53 PRINT"■":POKEV,0:FORN=0TO240STEP20:POKEW1,0:POKEW1,129:POKEH1,N
54 POKEL1,N+10:POKEA1,56:POKEW2,0:POKEW2,33:POKEH2,N:POKEL2,N+10:POKEA2,56
55 POKEW3,0:POKEW3,129:POKEH3,N:POKEL2,N+10:POKEA3,56
56 POKEV+24,20:POKE53281,0:POKE53280,0:PRINT"■■ FORMULA ONE ■ : WILLIAM & SIMON
 FONG"
57 PRINT"■■■■        YOU'VE COMPLETED THE GAME!":PRINTTAB(15)"■■■■CRASH:"L
58 GOTO73
59 D=-1:IFRND(1)<.6THEND=1
60 POKEW1,0:POKEW1,129:POKEH1,43:POKEL1,137:POKEA1,190
61 POKEW2,0:POKEW2,33:POKEH2,108:POKEL2,223:POKEA2,190
62 POKEW3,0:POKEW3,33:POKEH3,43:POKEL2,137:POKEA3,12:FORN=100TO200:X=X+(D*4)
63 IFX<118THEND=1
64 IFX>185THEND=-1
65 POKEV,X:POKEV+1,N:N=N+2:X=X+4:POKEV,X:POKEV+1,N:X=X-4:N=N-2:NEXT
66 X=150:POKEV,X:POKEV+1,100:L=L+1:IFL=3THEN69
67 POKEV+31,0:PRINT"■":OS=0:CR=0:POKE53280,5:POKE54272,18:POKE54273,1
68 GOTO27
69 POKEV,0:POKEV+24,20:POKE53280,0:POKE53281,0:PRINT"■"TAB(18)"■SORRY!"
70 POKE54272,37:POKE54273,17:POKE54272,0:POKE54273,0:POKE54278,0
71 POKE54296,15:POKE54277,236:POKE54278,236:POKE54276,33
72 POKE54273,7:POKE54272,53:PRINTTAB(8)"■■■YOU CRASHED THREE TIMES!"
73 PRINT"■■■■    PRESS FIRE BUTTON FOR ANOTHER GO"
74 FORN=240TO0STEP-1:POKEW1,0:POKEW1,33:POKEH1,N:POKEL1,N+10:POKEA1,56
75 POKEW2,0:POKEW2,33:POKEH2,N:POKEL2,N+10:POKEA2,56
76 POKEW3,0:POKEW3,33:POKEH3,N/2:POKEL2,N/2:POKEA3,56:NEXT
77 FORN=0TO240STEP10:POKEW1,0:POKEW1,33:POKEH1,N:POKEL1,N+10:POKEA1,56
78 POKEW2,0:POKEW2,33:POKEH2,N:POKEL2,N+10:POKEA2,56
79 POKEW3,0:POKEW3,33:POKEH3,N/2:POKEL2,N/2:POKEA3,56:NEXT
80 AA=PEEK(56320)AND16:IFAA=0THENRUN
81 GOTO80
82 DATA 15, 255, 192, 3, 255, 0, 4, 220
83 DATA 64, 4, 220, 64, 4, 220, 64, 3
84 DATA 239, 0, 3, 155, 0, 3, 87, 0
85 DATA 3, 171, 0, 3, 87, 0, 3, 223
86 DATA 0, 3, 255, 0, 3, 255, 0, 3
87 DATA 255, 0, 0, 252, 0, 0, 252, 0
88 DATA 1, 185, 0, 1, 49, 0, 1, 49
89 DATA 0, 0, 252, 0, 0, 0, 0
```

REFERENCE LIBRARY

This month's look at the books which should be filling every self-respecting Commodore user's shelves.



Book Title:
Data Handling on the Commodore 64 Made Easy
Author:
James Gatenby
Publisher:
Granada
Price:
£5.95

Mr. Gatenby's book appears to be yet another introduction to the BASIC programming language, this time under the guise of introducing the reader to the world of data processing. It starts by allaying the reader's fears that any knowledge of mathematics is needed to write data handling programs and promises that, with this book to guide you, you will be able to design programs to store large amounts of data, display the data on the screen in an attractive and readable way, search the data for particular items and print out the relevant information and sort, update and amend the data, all at a speed to make manual systems redundant.

The book introduces the reader to the most basic tenets of computing: terms such as 'programs', 'microchips', 'cursor' or program commands such as 'LOAD', for example, are explained. The greater part of the rest of the book seems to be devoted to teaching the reader BASIC in conjunction with using BASIC to create data handling programs such as a telephone directory program, one of the examples given by Mr. Gatenby. The latter chapters cover the areas of programming more relevant to data processing such as searching, sorting, modules, menus and file handling. The book concludes by advising the reader on how to extend his system by adding to it peripherals such as a disc drive or printer.

To conclude, this book is a clear, concise introduction to the world of processing data to produce useful information. However, I think it is tailored really for the novice programmer: anybody with a relatively sound knowledge of BASIC who wants to explore the Commodore 64's data handling capabilities should turn to the latter chapters of the book.

Book Title:
The Last VIC-20 Book in the World
Author:
Tony Noble
Publisher:
Sigma Technical Press
Price
£5.95

TO MANY CHILDREN (and adults, too) the words 'learning' and 'fun' aren't usually synonymous. Mr Noble sets out to crush this idea with his book which claims to make education fun by allowing his readers to learn through game-type situations. Children may learn in their own time, unpressurised by the classroom atmosphere.

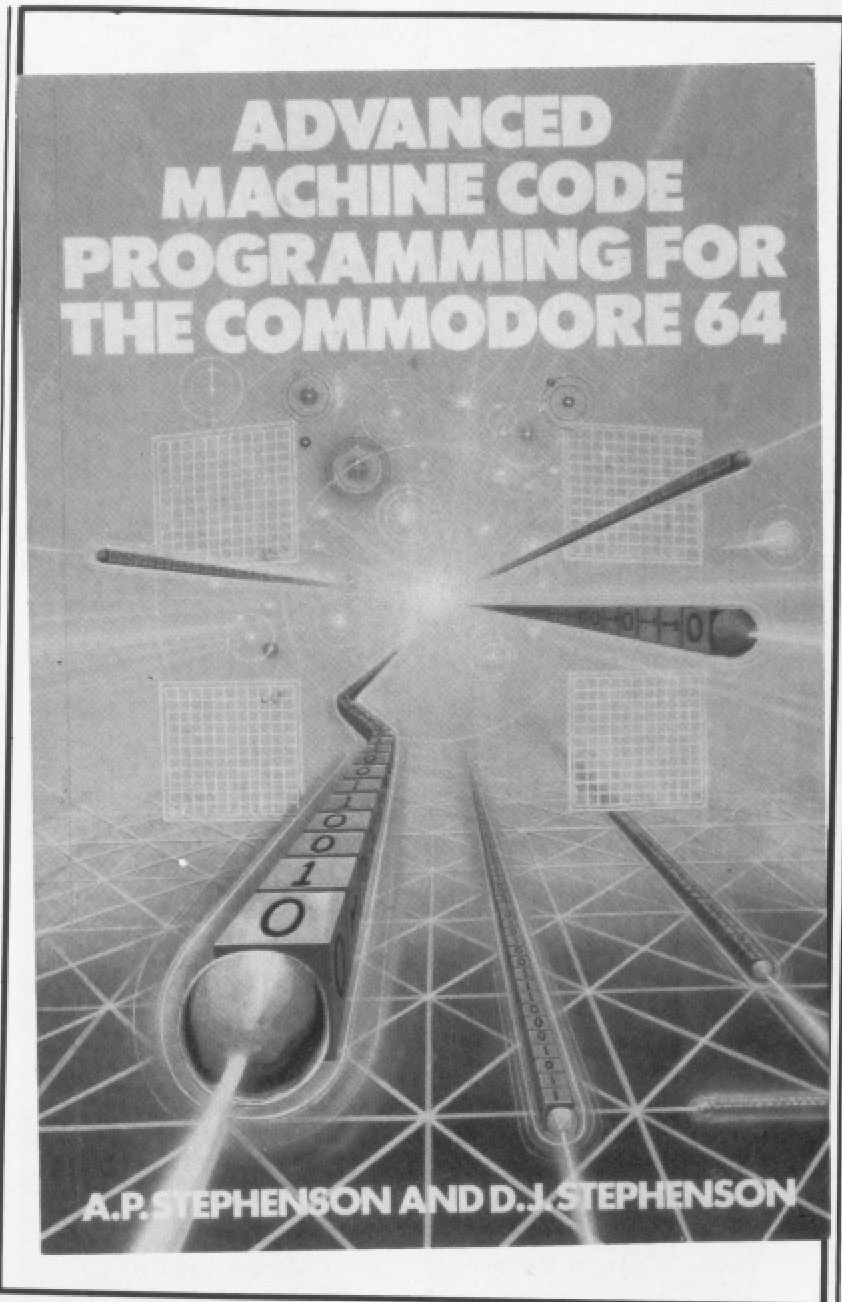
The games cover such diverse aspects as arithmetic, algebra, spelling, geography and French. Titles such as Galactic Adventure and Nessie the Loch Ness Monster may deceive the reader into thinking this book is jam-packed with run-of-the-mill computer games but, behind a facade of space ships and monsters, the reader is encouraged to improve his geometry or logical thinking. Other games included are 10 Green Bottles (to test your algebra), Word-Find (to improve your vocabulary, spelling and letter manipulation) and Crack-the-Code (to test your logic). Purely for younger children, exercises such as Odd-One-Out (to test visual discrimination)or counting games are provided. Improve your French with Parlez-vous Francais or your multiplication with the brain-twisting Multi Marathon. Less alluring, and more self-explanatory, titles

include Geography and U.S.A. Quiz.

So, say 'Goodbye' to stuffy classrooms and dusty textbooks and delve into 'The Last VIC-20 Book in the World'.

Book Title:
Commodore 64 Disk Companion.
Author:
David Lawrence and Mark England
Publisher:
Sunshine Books
Price:
£7.95

THIS IS A BOOK THAT IS long overdue. It seems that Commodore have no intention of updating their own manual, and the sheet of corrections they promised to send me have not yet shown their face. Sunshine books have done it. A well written, easy to understand book which takes away the mystery of relative, user, and program files. It is easy to just dip in and find out what you need and try it. It usually works. The demo programs are not quite up to the mark though and I feel that some users may get bogged down when they try to step through what each program is doing, but their explanations make up for that. If you have a 1541 drive and are still having problems then this book will save your sanity. A little expensive at 7.95 but I think I spent more than that on aspirin when using the Commodore manual.



Book Title:
Advanced Machine Code Programming for the Commodore 64
Author:
A.P. Stephenson and D.J. Stephenson
Publisher:
Granada
Price:
£7.95

THIS IS NOT SO MUCH A book for the machine code beginner but for those of you with a sprinkling of machine code knowledge which you wish to build upon; the authors do claim, however, that, so long as you've got to grips with BASIC and are prepared to put in a lot of hard work, this book may be used as an introduction to machine code. Most of the material contained in the book may be used with the 6502 processor which is similar to the Commodore's 6510A processor.

Throughout the book, the text is illustrated with many examples including full listings accompanied by hex dumps. An assembler is needed to master machine code; the MIKRO 64 assembler has been used to develop the programs listed in this book. Each chapter is concluded with a succinct and useful summary of the chapter for quick and easy reference and, so as to check your progress as you plough through the book, short tests (with answers at the back of the book) are provided.

The book starts by assessing BASIC, compilers, interpreters, ROMs, RAMs and other general topics and moves on to discuss the 6502/6510A microprocessor, various modes, entering and assembling code. It then covers the area of programming in which the advantages of machine code over BASIC really come to the fore, namely sort routines, high-resolution graphics, sound and, finally, an outline of TTL logic for those readers whose interests extend beyond mere programming.

Not so much a straightforward introduction to machine code, this book is rather the serious programmer's guide to mastering machine code language on the Commodore 64.

Book Title:
Commodore 64 Disk Systems and Printers
Author:
Ian Sinclair
Publisher:
Granada
Price:
£9.95

THE BOOK'S OUTWARD appearance is bright and eye-catching with a 'computers in space' design adorning the front cover. It provides the Commodore user, in particular the disc system novice, with information on disc systems and printers available for use with his machine. As well as assessing the disc drive, the 1541, and disc systems peculiar to the Commodore 64, the book covers the commands to be used with the disc systems, primarily LOAD and SAVE, various disc utilities and, in greater detail, the filing actions, an integral part of business and database applications for which disc systems are largely required. For those of you who do not merely intend to use your disc system as a means of storing programs and data, but wish to master the techniques of copying and deleting files, backing-up discs, writing machine code disc routines or reading data from damaged discs, Mr. Sinclair's book provides you with the knowledge to fulfil these highly important functions. A whole chapter is devoted to a database type program which comprises a long listing followed by a detailed explanation. The last chapter offers information on the various printers available for use with the Commodore 64 and summarises the ways in which (and the success with which) they fulfil their function.

To conclude, the author believes, quite rightly, that

disc systems are a must for the serious programmer due to the greater memory capacity they append to the computer. "Commodore 64 Disk Systems and Printers" includes a few (maybe too few) clear and succinct diagrams. The listings throughout the book are made easier to follow by replacing the usual indistinctive Commodore graphic symbols with CHR$(n) commands. A selection of the usual appendices are provided covering, for example, Random Access Files, lists of commands and hardware suppliers.

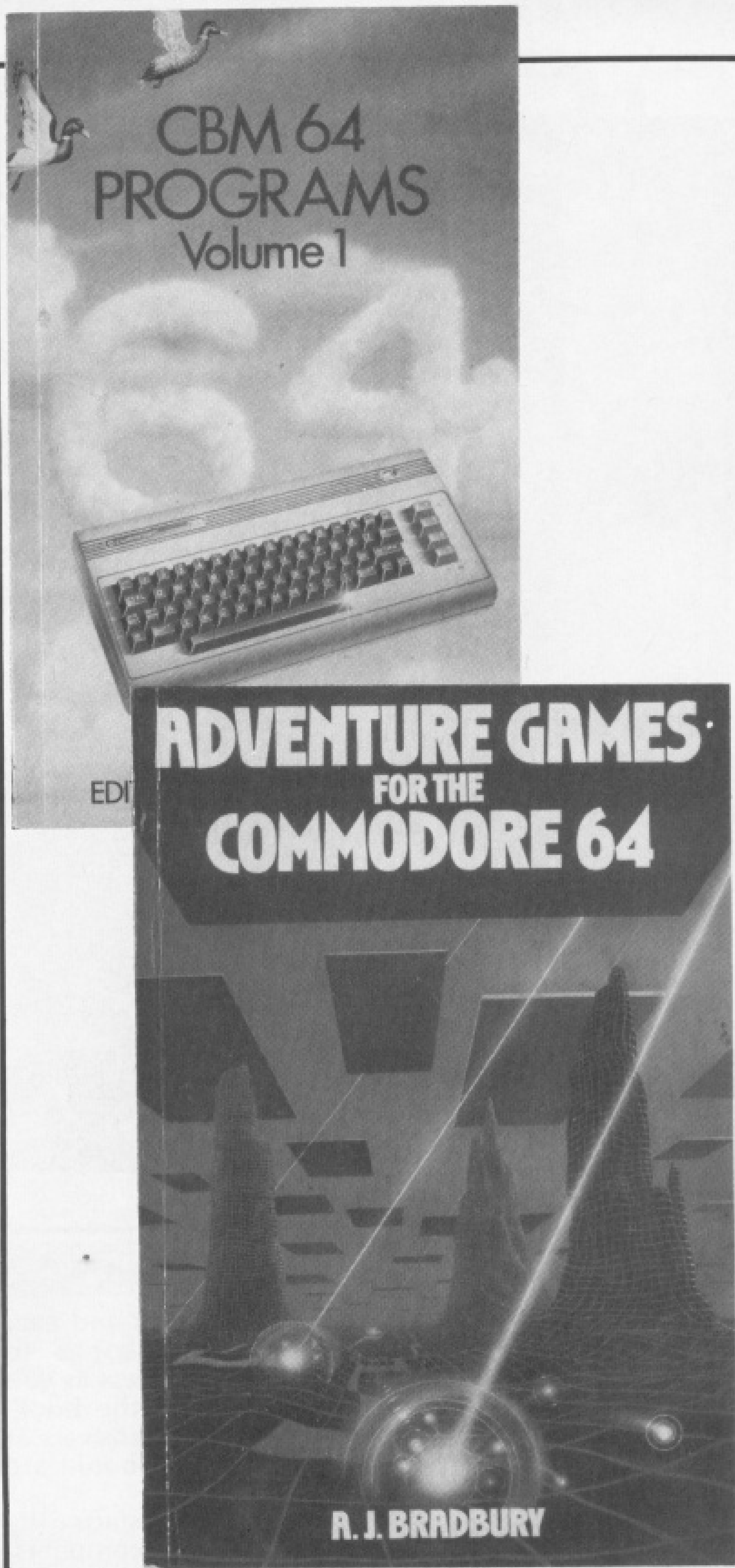
Book Title:
CBM 64 Programs Volume 1
Programs By:
Richard Franklin
Edited By;
Nick Hampshire
Publisher:
Duckworth Home Computing
Price:
£6.95

THE COMBINED EFFORTS of Mr. Franklin and Mr. Hampshire have produced a wide selection of programs to be keyed into your CBM 64. Games, utilities, music, graphics, sprites and user defined characters, and functional programs are all covered in this book.

From an introduction to graphics characters and machine code routines, the book allows you to boldly go where probably many a Commodore 64 user has been before — into space, this time with a fairly comprehensive version of Star Trek. Other games included, none of them highly original, are Hangman, Landmine, Fruit Machine, Car and Maze.

But it's not all fun and games. A useful section is included on Hi-Res graphics containing such gems as programs to plot bar charts in multi-colour, to display a three-dimensional graph in standard Hi-Res and to display the time as input from the user in the form of a 24 hour digital clock. Keyboard Synthesizer allows you to exploit the music capabilities of the Commodore 64 by using the keyboard as notes. Other utilities include programs to change the reserved words of the Commodore 64, to convert machine code programs to decimal data statements and add them to the program and to store personal information in the form of addresses or diary entries in the program. The book ends on a note of adventure with Will O' The Wisp.

You will need a degree of care and patience to enter some of the lengthy listings contained in this book. Having crossed that hurdle, although most of them have been seen somewhere before, you will find here a broad selection of programs, some useful, some entertaining, for your Commodore 64.



Book Title:
Adventure Games for the Commodore 64
Author:
A.J. Bradbury
Publisher:
Granada
Price:
£6.95

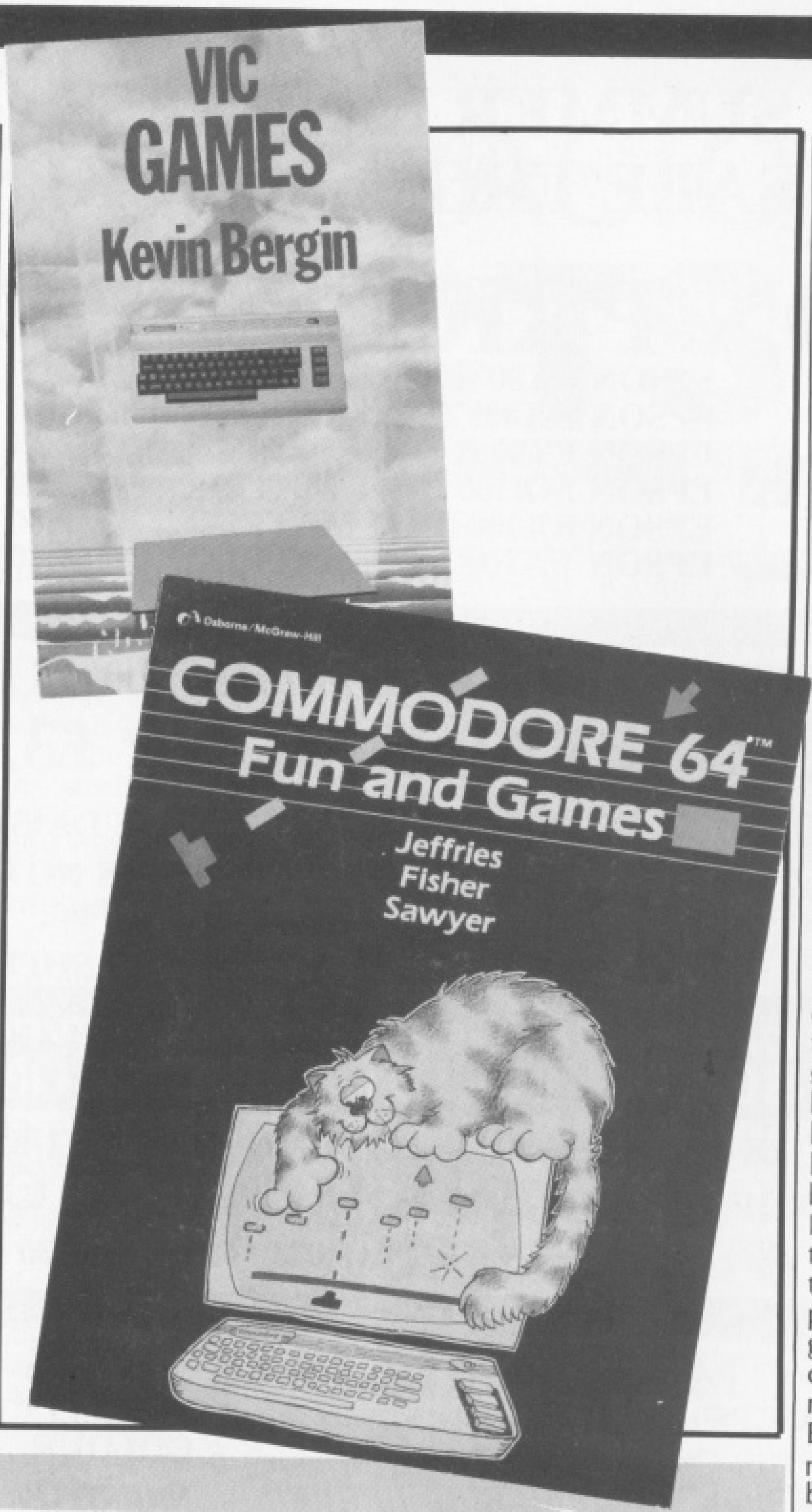
HAVING EXHAUSTED THE software industry's stock of adventure games for the Commodore 64 and consequently having realised that, with the programming know-how and one or two bright and original ideas, you could do better yourself, here is the book to get you started. Not only novices but even experienced programmers wishing to make their adventure programs a viable financial concern should find this book useful.

The book commences with a potted history of the computer adventure program and lists, and expands upon, the most salient points to remember when creating your own adventure. Before churning out reams of code, the adventure story programmer has to have a story; the book goes on to guide the reader in devising a suitable storyline and in creating the characters to participate in the adventure. The reader is shown, step by step, how to build his adventure program adding graphics, words and sound until the completed adventure game eventually takes shape. Many examples and sample programs are included throughout the book. The book ends by predicting the adventure game of the future.

Although this book claims to cater for the beginner, a total novice to BASIC may have to look elsewhere for an introduction to the language before tackling some of the code in the latter half of the book. This criticism aside, Mr. Bradbury has produced a relatively original volume in a market where the subject matter of the literature outputted is usually highly repetitive.

Book Title:
Business Applications
Author:
James Hall
Publisher:
Sunshine Books.
Price:
£5.95

THIS BOOK IS IN EFFECT one long program which has parts that interact with the rest. If you can dig out the parts that are helpful to you it has some very effective and interesting techniques. The sub-routines are useful though there are better ones available. The book does what it says but not in quite the way I like, but others rave about it (but then I like B. Cartland). At 5.95 it is a good buy and one that would have its uses.



Book Title:
Vic Games
Author:
Kevin Bergin
Publisher:
Duckworth
Price:
£6.95

ALTHOUGH ENTITLED VIC games, this book is a collection of games and utility programs for the VIC 20. The games are accredited with being 'exciting and interesting'. How accurate an assessment this is cannot be ascertained without putting finger to keyboard and actually testing the games. Each game is, however, clearly explained and set out with descriptions and program structures preceding the listings.

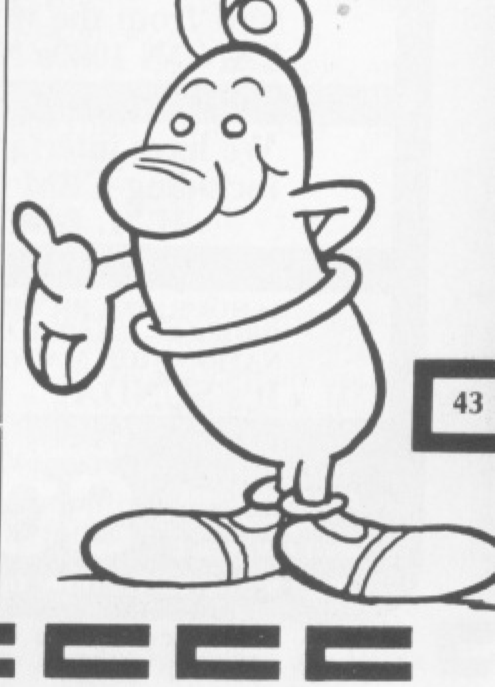
Some of the games, such as Punter, where you have to escape the claws of death while on an excursion to the Post Office and a pretty skilful golf game (simply called Golf), appear to be more interesting than many of the volumes of games around for the Commodore range of machines. Others such as Air Attack or Invaders sound all to familiar. A couple of adventure games are also included: Grobbit, a mini-adventure maze-type game or Agent, the object of which is to find the microfilm and pass it over to your contact while surviving the deadly attackers. A few useful utilities are also provided: Code Creator which creates data statements from Machine Code routines and places them in user programs in BASIC lines or Tape Search which saves time by searching for and loading programs.

Maybe more thought could have gone into the order of the book's contents — placing the utilities at the end of the book may have made more sense than interspersing them with the games. But, on the whole, Kevin Bergin has produced a good, broad selection of games for the VIC 20.

Book Title:
Commodore 64 Fun and Games
Author:
Ron Jeffries, Glen Fisher and Brian Sawyer
Publisher:
Osborne/McGraw-Hill
Price:
£9.95

HERE IT IS FOLKS! — A BIG, bright and bumper book of fun and games for your Commodore 64. These 35 games exploit the 64's special features such as colour, sound, sprites and graphics and, since all you need to do is copy what you see in front of you, you don't need to know one end of a BASIC instruction from another. Before launching into the games, the book wets your appetite with a set of small programs to get you used to the notation used in the program listings and the colour and graphics capabilities of the Commodore 64.

The games cover a broad spectrum ranging from the common-place, forgettable board-type game such as Dots, Spot, Reversi and Leap to the fast and furious Fire, where you have 3 minutes to extinguish a fire, and Godzilla, where the might of the Japanese armed forces are attempting to catch Godzilla. Treasure hunts loom large with Miser or Dive, where the object is to receive treasure from a sunken ship. A sense of humour is required for the ludicrous Bat where you lead the life of a bat bouncing off walls, etc., or Lawn where the object is, as the title may imply, to mow the lawn as quickly as possible. Try your hand at gambling with Bjack or Bets, create music with Piano or race either to the top of Everest with Everest or merely to the top of the board with Bonzo. And they're many more besides packed into this entertaining and easy-to-follow, albeit not entirely original, book of fun and games.

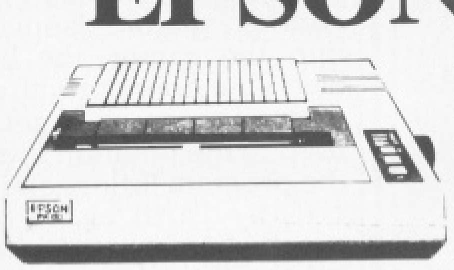


ORDER TODAY
PRINT TOMORROW
24 HOUR DELIVERY

SUMMER MADNESS SALE FROM SCI(UK)

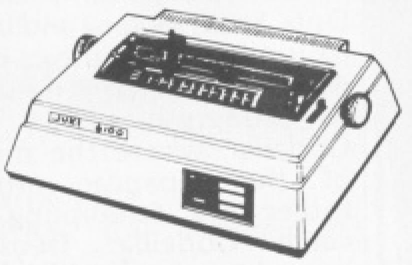
OPEN 7 DAYS A WEEK

EPSON PRICE SPECIALS



EPSON RX80 (DOT MATRIX).... ~~£249.00~~ £199 + VAT = £228.85
EPSON RX80FT (DOT MATRIX). ~~£285.00~~ £229 + VAT = £263.35
EPSON FX80 (DOT MATRIX).... ~~£438.00~~ £324 + VAT = £372.60
EPSON MX100 (DOT MATRIX).. ~~£475.00~~ £355 + VAT = £408.25
EPSON RX100 (DOT MATRIX)... ~~£450.00~~ £385 + VAT = £442.75
EPSON FX100(DOT MATRIX)... ~~£569.00~~ £499 + VAT = £573.85

DAISYWHEELS...*at an incredible new* LOW PRICE!



JUKI 6100......*just* £329 + VAT = £378.35

20CPS: BiDirectional & Logic Seeking
10, 12, 15 & Proportional Spacing
Wordstar Compatable
2K Buffer: 13 inch Platen
Underline: Backspace + Lots more
Centronics Interface Standard

THE DAISYWHEEL THAT HAS NO COMPETITION

OPTIONAL RS 232 TRACTOR AND SHEET FEEDER

We will match any Genuine Price Advertised—
SCI(UK) IS NEVER BEATEN ON PRICE

MANY MORE PRINTERS AVAILABLE - 1000s of SCI(UK) BARGAINS



***send now for the* FAMOUS SCI(UK) Catalogue**



for the cheapest prices telephone 0730 68521 *or* 0730 68522

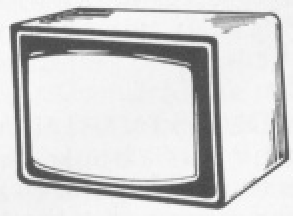
MORE SCI(UK) BARGAINS

SHINWA CP80....£179.00 + VAT = £205.58



Friction and tractor feed as standard
80cps
Bi-directional logic seeking
13 x 9 dot matrix giving true descenders
Sub and superscripts.
Italic printing and auto underlining
Condensed, emphasised, expanded and double strike printing (can be mixed in a line)
Parallel interface fitted as standard

FIDELITY 14" COLOUR MONITOR & COMPOSITE VIDEO
£189.00 + VAT = £217.35



New from the world famous CANON Company
CANON 1080a NLQ DM best value ever at.............................£289.00 + VAT = £322.35

We have interfaces for all types of computers, including CBM 64, VIC 20, APPLE, TRS 80, IBM, BBC, SPECTRUM, QL, etc.

Access · MasterCard · AMERICAN EXPRESS

24 HOUR SECURICOR DELIVERY £9.50 plus VAT ● BANKERS ORDERS, BUILDING SOCIETY CHEQUES, POSTAL ORDERS — SAME DAY DESPATCH ● ALL ORDERS COVERED BY THE MAIL ORDER PROTECTION SCHEME
NATIONWIDE MAINTENANCE CONTRACTS ARRANGED ● EDUCATIONAL DISCOUNTS VERY WELCOME
It's SUNDAY - Do you realise you can order NOW - We are open 7 Days a Week.

DEALER ENQUIRIES WELCOME WRITE FOR DETAILS

RANGEKEY LTD. Trading as SCI(UK)

SCI(UK)

SCI (UK) FREEPOST (No stamp needed)
PETERSFIELD HANTS GU 32 2BR

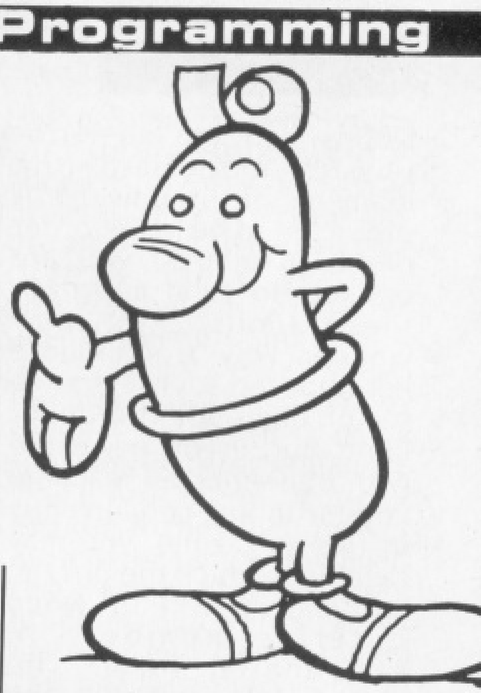
0730 68521
0730 68522

EXPORT ENQUIRIES
NO TAX
DELIVERY AT COST

Large showrooms now open at 12 High Street, PETERSFIELD, Hants. GU32 3JG Telex 86626 MYNEWS G

In part 2 of their series on BASIC, A.P. and D.J. Stephenson discuss variables, assignments and operators.

THE BASIC FACTS



IN PART 1 OF THIS SERIES, we discussed variable names. Before describing the different types of variable we ought to make sure we know exactly what is meant by 'variable'. This word, and many others used loosely in everyday speech, take on a more restricted and precise meaning when they are used by programmers. It would seem a hair splitting distinction to point out that the terms 'variable' and 'variable name' are not quite the same. **Variable names**, such as AB, BS, or C1 are really symbolic addresses, chosen by the programmer, to hold chunks of information. If we write,

A2 = 34556983

we are telling the computer that an area in memory is to be known henceforth as A2 and that, for the moment, this area is to hold the number 34556983. We say, 'for the moment' because it is quite possible that we will, later on in the program, make A2 hold a different number. In other words, the contents of the memory area can be **varied** — hence the name.

We should realise that when we name an area in memory, as A2 or ZF, the BASIC interpreter in ROM is responsible for allocating the group of machine addresses in memory corresponding to the variable name we have chosen. Such machine addresses are transparent to the programmer and virtually of no interest. However, it is worth mentioning that several machine addresses are reserved to hold each variable irrespective of its magnitude. A single memory cell is only eight bits 'wide' which, if you have experience in binary arithmetic, should tell you that the highest absolute number which can be stored is only 255 decimal.

Three kinds of variable are distinguished by the BASIC interpreter and they must not be mixed up.

Integer variables

An integer is a whole number, positive or negative. It has no decimal point anywhere. For example, 3, 468, 58694, 1000 are all integers. The largest possible integers allowed in the Commodore 64 are 32767 or -32767. To inform the interpreter that the variable is to hold only integers, the character '%' must be written after the variable name. Examples: A%, B3%, SD% are all integer variables.

It is no good trying to store 34.6 in A%. If you do, the computer will ruthlessly **truncate** downwards to the nearest lower integer. For example, if we write A% = 34.6, the contents of A% will be truncated to 34. The fraction part is lost. On the other hand, if we write A% = –34.6, the contents of A% will be –35. If you think this last result conflicts with our earlier remark, remember that -35 is considered (by mathematical types) to be a **smaller** number than –34. If I have a deficit of 34p, I am better off than if I had a deficit of 35p.

Floating point variables

A **floating point** variable can have a decimal point somewhere and can be positive or negative. Numbers and measurements in real life are seldom integers, so floating point numbers are often known as 'real' numbers to distinguish them from subsets such as integers. No special suffix is needed after the variable name if it is to hold floating point numbers. For example, A1, ZD, R are all considered by the interpreter to be floating point and they can all hold numbers such as, 200.46 or -456.025.

There is an awkward little quirk which needs sorting out regarding integers and floating point. Although we know that integer variables can only hold integers, it is not immediately evident that floating point variables can also store integers. This is because floating point numbers are real numbers and real numbers include integers! Thus, there is nothing agains writing A = 5 or B2 = 500 or, come to that, C = 5.0, even though the '.0' is technically redundant.

Summarising, a floating point variable can hold all types of numbers including integers but an integer variable can only hold integers. When you are a newcomer to BASIC, you may find it safer and less complicated to use only floating point variables but, as your experience grows, you would be well advised to use integer variables wherever possible — they execute a little quicker after a RUN and they take up a little less space in memory.

Very large and very small numbers in physics and other sciences are represented in 'exponential form' in order to cut down on the number of noughts. For example, the electrical charge on the electron is so small that it would require 18 noughts after the decimal point before the first significant figure. Written out in full, we would get the following revolting mess:

0.000000000000000000159 coulombs.

Even this is abbreviated to three significant digits. Besides being error prone for humans, this clumsy notation would be beyond the resources of the Commodore 64 because we are limited to nine digits of precision including leading zeros after the decimal point. To overcome the obstacle presented by large and small numbers, we can use exponential notation in floating point numbers. This notation consists of two parts:

- The significant digits terminated by the letter E
- The power of 10 multiplier

Example: 310 can be written as 3.1E2 which means 3.1 multiplied by 10 raised to the power 2. In ordinary maths, this would be written as 3.1 x 10 ^ 2.

Example: 0.00031 can be written as 3.1E-4.

As a final example, we return to the charge on the electron. This now becomes:

1.59E-19

(Note carefully that the negative exponent is always one more than the number of noughts before the first significant digit).

There are certain upper limits to be observed. If you exceed them, you will get a nasty message from the computer. These limits are as follows:

+ or – 1.70141183E+38

+ or – 2.93873588E-39

Unless you are an astrophysics enthusiast (the diameter of the universe is predicted to be in the order of 10ˆ76 metres) you are unlikely to even approach these limits. You may wonder why Commodore has imposed such a strange set of digits for the upper limits but, if you persevere with our series on machine code (running concurrently in this magazine), you may be able to crack the puzzle.

Be very careful when entering numbers in exponential form. The power of 10 multiplier (the exponent) is more important than the significant digits (the mantissa). If you are four out in the mantissa you could be mildly out in your calculations. If you are four out in the exponent, the mistake will border on the catastrophic. The previous examples of exponential form may have given the impression that the mantissa must always be a single digit followed by a point. This is not so. There are various ways of fiddling around with the mantissa and the exponent. For example, 2.456E5 can be written as 24.56E4 or 245.6E3 or as 2456E2 because all three forms represent the same absolute number. It is just a case of juggling with the mantissa and exponent. As you move the point, a corresponding change must be made to the mantissa. Not only can you enter numbers in exponential form, the computer automatically prints out in this form if the number is *less* than 0.01 or *greater* than 999999999.

String variables

A string variable can hold virtually anything. Although a string can hold a number, it can also hold letters, punctuation and special characters. To inform the computer that the variable is to hold strings, the name must end in the dollar sign '$'. For example, A$, D3$, SF$ are all string variable names. Although the mixture doesn't matter, the total number of characters in one string variable must not exceed 255.



When we put something into a string variable, it is vitally important to observe the so-called 'matching' rule. This simply means that both sides of an assignment must be string variables or in equivalent string form. Before going into details of mis-matching we must remember that when we wish to assign a string of characters to a string variable, they must be enclosed in double quotes. For example, suppose we wish to store the following message in a string variable named M$:

```
DANGER!   UNEXPLODED
BOMB
```

We must enclose the message in double quotes:

```
M$ = "DANGER ! UNEX-
PLODED BOMB"
```

Note that there is no mismatch because both sides are strings.

Here are some legal assignments,

```
A$ = "    "
B$ = "WELCOME"
C$ = B$
```

Here are some illegal assignments which will cause a mismatch error:

```
A$ = WELCOME
B = "GLOOM"
C$ = K
D$ = 345
```

Notice the last mismatch which appears to indicate that we can't store numbers in a string variable. However, we can store numbers in strings providing we enclose them in double quotes. For example, A$ = "345" is quite legal and 345 will indeed be stored in A$, but you can't do normal arithmetic on numbers held as strings. It will never be treated by the arithmetic circuits of the computer as a number — it will be treated as three ordinary characters.



Concatenation of strings

Although normal arithmetic cannot be performed on strings it is possible to use the '+' sign between strings in order to join them together into a single large string. This is known as **concatenation.** For example, study the following segment:

```
100 A$ = "CONSER"
110 B$ = "VATION"
120 C$ = A$ + B$
```

The string variable C$ will now hold the word CONSERVATION. If the last line was changed to:

```
120 C$ = B$ + A$
```

then C$ would hold VATIONCONSER. This illustrates vividly that concatenation is very much different from arithmetic addition, even though the same '+' sign is used. In normal arithmetic, 5+3 is the same as 3+5.

Concatenation does not allow an escape route for the 255 character limit. For example, suppose A$ contains 200 characters and B$ contains 200 characters. Writing, C$ = A$+B$ in an attempt to break the rule will end in failure (and an error message).

Assignments

When we write A = B we have assigned the value of B to A. Assignments are the most common of all computer operations so it is important to examine some of the possible pitfalls. The rules are as follows:

1. The variable on the right of the equals sign is copied into the variable on the left.
2. The previous contents of the left-hand variable are lost because the new contents have *overwritten* the old.
3. The contents of the right-hand variable remain unchanged.

As an illustration of the rules, suppose that before the assignment, A contained 50 and B contained 30. After A = B, both A and B will contain 30. These rules are simple but it is so easy to get the assignment the wrong way round. Remember — the left-hand variable will receive the result of the assignment. As a self-test exercise, study the following programming segment:

```
100 A = 30
110 B = 50
120 C = 70
130 A = B
140 C = A
```

The contents of the variables after the above is executed are as follows:
A = 50   B = 50   C = 50

Arithmetical assignments and operators

The left-hand side of the equals must be a single variable but the right-hand side can be any legitimate **expression,** usually arithmetic in nature.

The kind of operation performed depends on the operator. There are three classes of 'operator', but, for the moment, we are only interested in the algebraic class. There are six of them, and although they are listed at the back of the Commodore User Manual supplied with the machine a few extra comments may help.

The '+' operator is used for addition when used between two variables or numbers.

```
Example: C=A+B or C=30+50
```

The right hand side is the expression. The machine evaluates the expression and places the result in C.

The '−' operator is used for subtracting one number or variable from another. The rule is the same as in normal arithmetic, the quantity on the right is subtracted from the quantity on the left. The '−' sign can also be used before a variable to indicate it is a negative number.

Example: C=A–B

It is worth emphasising again that the quantities on the right of the equals sign are not altered in any way — it is only the single variable on the left of the equals sign which has its contents overwritten by the result of the operation. In the example above, although C will finally receive the result of A–B, the contents of A and B have not been altered in any way by the arithmetic process. This is because the BASIC interpreter takes copies of the variables for calculating results.

The operator '★' is used for multiplying two numbers or variables together.

Example: C=A★B

We mentioned in part 1 of this series what happens if we want multiplication of A x B but forget, due to common usage, to include the asterisk between them. In normal algebra, C=AB implies multiplication but the computer will not multiply because AB is a legitimate variable name.

The operator / is used for dividing one variable or number by another.

Example: C=A/B

C will contain the result of dividing A by B, just as in normal algebra.

The ^ operator is used to raise a number or variable to a power. It is called **exponentiation.**

Example: C =A ^ 3

This is the same as writing C=A★A★A but is much cleaner and quicker.

Be careful if your variables are large and the power is large. The exponential operator has an enormous appetite for magnitude and it is not too difficult to produce overflow or underflow errors. The power can be positive or negative so we have to remember that a number too small can cause **underflow** errors although the error message from the computer will still say overflow error.

Example: PRINT 10^67

will certainly trigger off an overflow error and so will PRINT 10^–67.

Parenthesis

The characters '(' and ')' are used to indicate parenthesis or, using everyday language, 'brackets'. They act as a box to override the natural rules of **precedence** in the computer (see page 27 of the Commodore 64 User Manual). As recommended in Part 1, use them liberally



because the arithmetic in complex expressions can be very hard to follow if too much reliance is placed on precedence. Paranthesis are used exactly as they would be in ordinary algebra although, once again, it is easy to make a mistake by omitting the multiplication operator.

Example: D=A(B+C)

won't work as intended. It would with normal pencil and paper algebra but not in BASIC. It should be written D=A★(B+C).

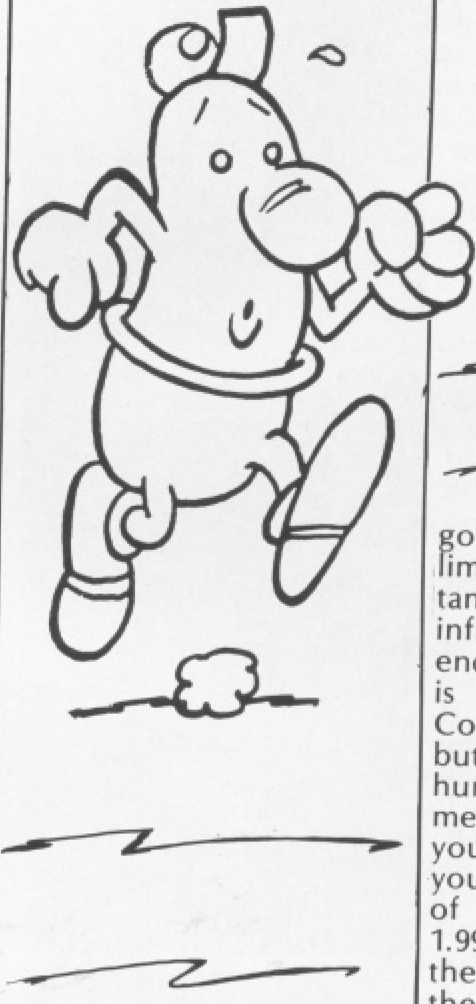
Another common source rce of error which can cause frustration is where parenthesis are used wrongly in division.

Example: D=(A+B)/C+D

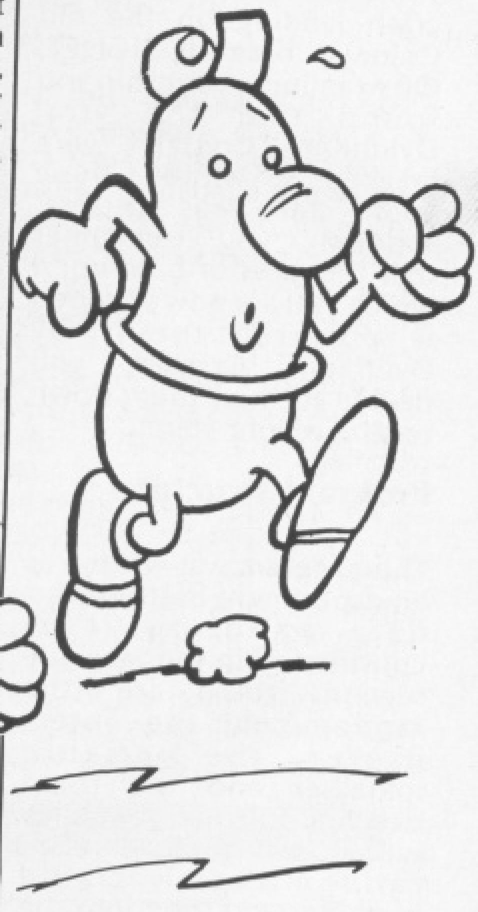
A+B is divided by C first and D is added afterwards. If you intend to divide by C+D then it should be written, D=(A+B)/(C+D).

Rounding errors

It is commonly supposed that computing arithmetic is dead accurate. For most practical purposes, the supposition is true. However, slight errors can creep due to the finite precision of the BASIC interpreter. According to Commodore, the arithmetic is correct to nine significant digits. Internal calculations are taken to ten digits but, before printing out the result, the last digit is dropped and the ninth digit rounded. If you are a newcomer to computing, don't be disillusioned by this. After all, how important is an error in the ninth digit? We should remember that in prehistoric days, when the slide rule was in constant use, we were lucky to get three significant digit accuracy.



Even logs, which were considered quite good enough for most technical subjects, only gave four figure accuracy. There can be a slightly more serious problem when using the special functions such as SIN (x), COS(x) etc. These functions are produced by special machine code subroutines using equations which are good approximations to the function over most of the range. However, the accuracy of trig functions deteriorates at the extreme end of the range. This again can be expected because trigonometrical functions in real life tend to



go a bit funny near the limits. For example, the tangent of 90 degrees is infinity so it is still an enormous number when it is close to 90 degrees. Computers do their best but, like us, they are only human! All this is worth mentioning if only to warn you not to worry too much if you expect, say, an answer of exactly 2 but you get 1.99999999. To try one of these funny answers, run the following,

```
100 PRINT 10^20
```

You will get 9.99999998E–21 which is near enough to the correct 10E20.

You are trapped on the old Colonial Base Signet, with a damaged spaceship. W.M. Newland has not made it easy, but can you escape?

SIGNET



While transporting some cargo through the perilous Garfray sector of the Galaxy you ran into a meteorite storm; your ship was hit and you were forced to make a crash landing on the old Colonial base of SIGNET. On examining your ship you found that the vital Dylithium Crystals were cracked — without these your craft was useless; suddenly you remember that the class of base upon which you are now stranded is powered totally by Dylithium Crystals, you decide to hunt them down to use in your ship. . .

Program structure

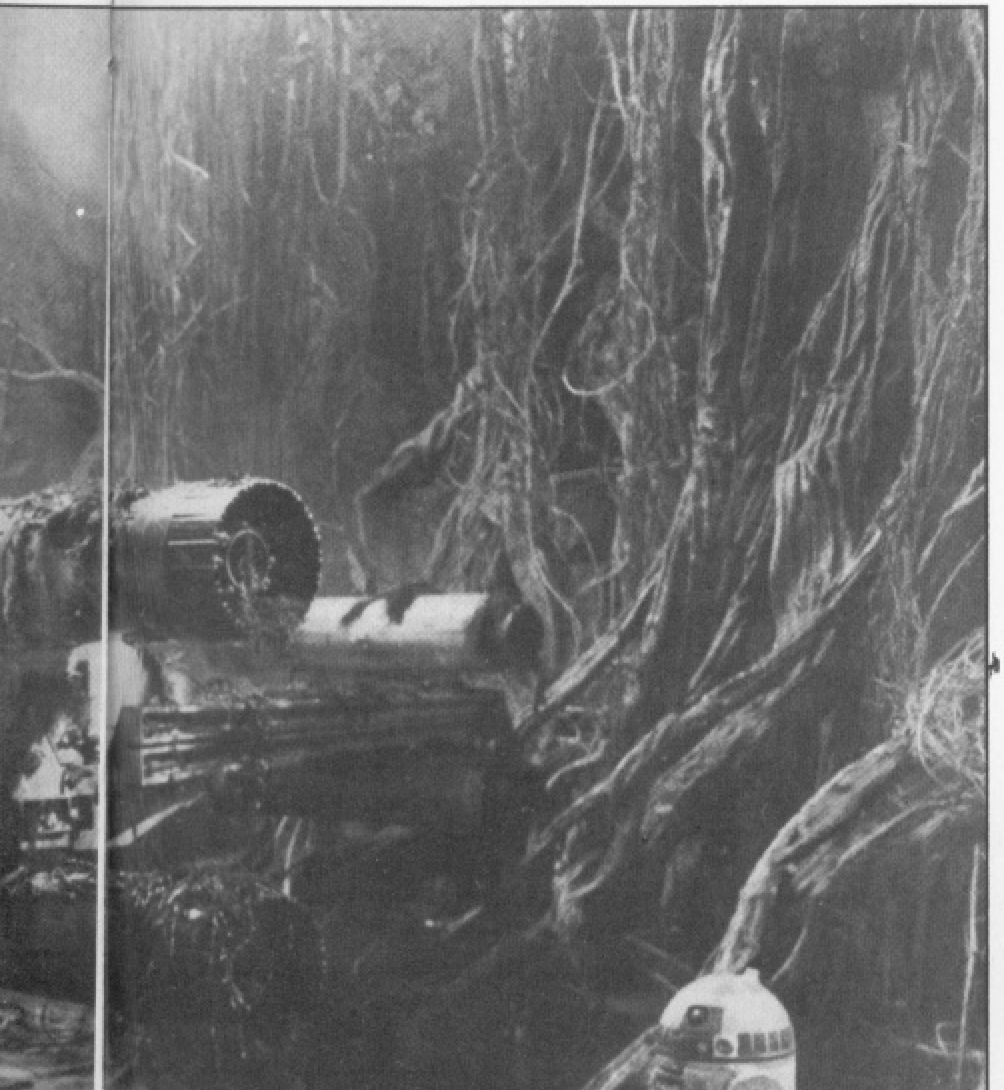
This program was written as an experiment in string and data handling it is constructed in four specific sections these are: the randomizer, the interpreter, the variable initialiser and the room descriptions. The computer will pass through the randomizer, variable initialiser and then into the room descriptions calling the interpreter as a subroutine from the last section.

The RANDOMIZER is handled by a short routine in lines 10-90, this is included to prevent the adventurer from jumping the early parts of the game if he inadvertantly gets killed.

The next section is the INTERPRETER this handles the user's entries and carries out his instructions; the first part of this resides in lines 100-235 and finds out which of the commands has been used and moves the computer to the required section. Line 240 is a default line which will print the message: "I DO NOT UNDERSTAND" if a command is used that is not within the games vocabulary (NB directions and "special" commands are handled at their specific locations). Having decided upon which command is being used the computer moves to the routine; these routines reside at:

TAKE: 260	CONNECT: 900
INSERT: 330	BOARD: 980
TYPE: 590	OPEN: 1130
WINCH: 640	DROP: 1300
SHOOT: 665	EXAMINE: 1500
FILL: 820	INV(entory): 1600
UNLOCK: 860	QUIT: 1800

The third section of the program is the VARIABLE INITIALISER; this sets up the major variables, namely: ITEM$ (room no, item no), ITEM (room no), ITEM (room no), INV$ (item no) & INV. The first ITEM$ holds the names of all the objects in the different locations; ITEM holds the number of items at each location, INV$ holds the objects you are carrying and INV holds the number of things you are carrying.



After this is a short ending routine and then the final section: ROOM DESCRIPTIONS, to explain this would be to solve the adventure so you'll have to experiment to find out how this works.

5	GOSUB to title page
10-90	randomizer
100-235	command decoder
260-1925	command subroutines
2000-2508	variable initialiser
2510-2580	ending routine
3000-4995	entry level room descriptions
5000-5950	lower level room descriptions
6000-7580	upper level room descriptions
10000-10200	title page

Line explanations.

This program should work on most PET computers (providing they have enough memory, approx 32K) although I have not tried it, just remove all POKEs and colour codes from within the quotes.

Text only

As this was a text only adventure I decided to use the neglected multi-colour background mode of the CBM 64. This allows the user to have up to four different background colours; if you wish to use this in your own programs here are the POKEs as they are not included in the USER MANUAL but they are explained fully in the PROGRAMMER'S REFERENCE GUIDE.

The POKEs are:

POKE 53265,PEEK (53265)or 64 — to turn on

POKE 53265,PEEK(53265) AND 191 to turn off

In this mode you do, however, lose the characters from code 64 onwards. Instead you have the same characters again with a different background colour.

Therefore:

Address 53281 effects 0-63
Address 53282 effects 64-127
Address 53283 effects 128-191
Address 53284 effects 192-255

These are all used in the same way as normal background POKEs.

I hope you find the program enjoyable and the POKEs useful.

Program

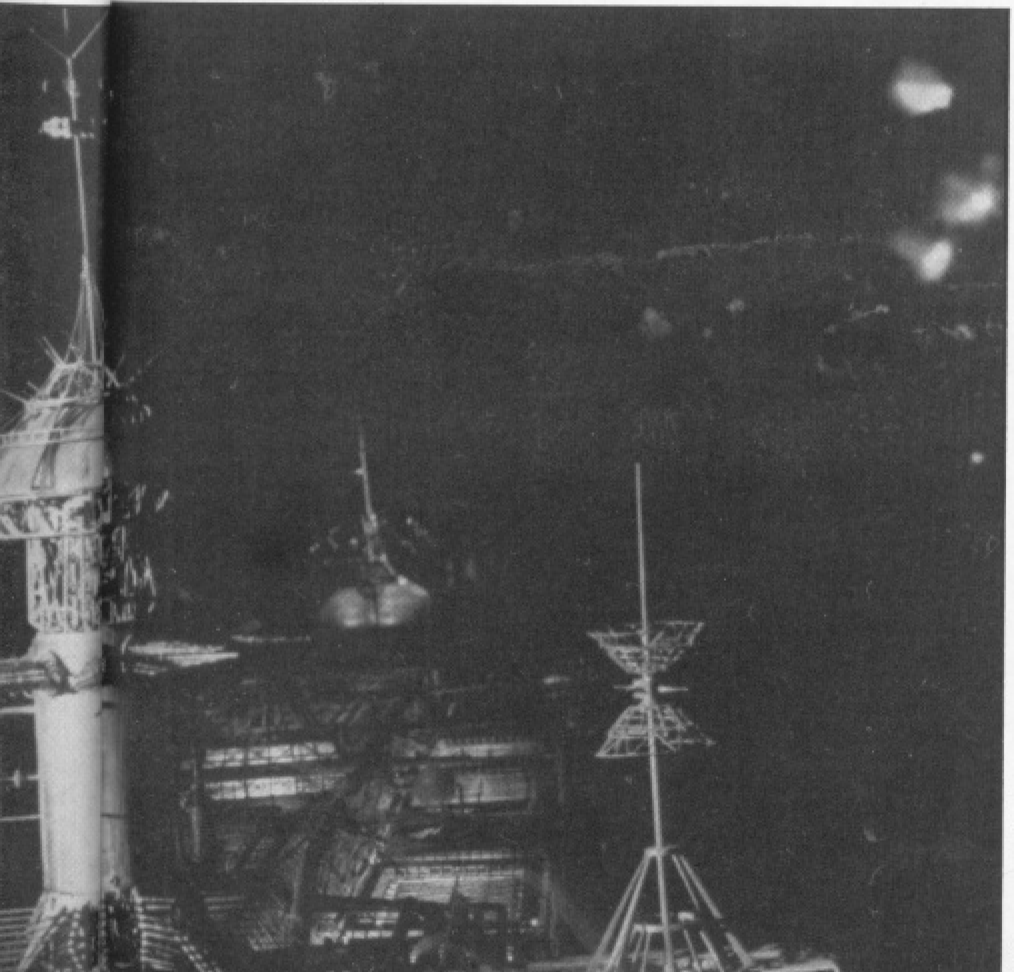
```
5 GOSUB10000
10 XX=INT(RND(TI)*6)
20 ONXXGOTO30,40,50,60,70
30 NN$="KARD A555":GOTO80
40 NN$="FRAG A649":GOTO80
50 NN$="PFILL Z990":GOTO80
60 NN$="RAIGL XX52":GOTO80
70 NN$="CAM U96X":GOTO80
80 FORCC=0TO6
81 C=INT(RND(TI)*(80-64))+64
82 CW$=CW$+CHR$(C):NEXTCC
83 CB=INT(RND(TI)*1000)+5000
85 DY=1000:DIMINV$(20)
90 GOTO2000
100 REM ENTRY
110 IFLEFT$(ENTRY$,4)="TAKE"THEN260:REM TAKE ROUTINE
120 IFLEFT$(ENTRY$,6)="INSERT"THEN330:REM INSERT ROUTINE
130 IFLEFT$(ENTRY$,7)="CONNECT"THEN900:REM CONNECT ROUTINE
140 IFLEFT$(ENTRY$,5)="BOARD"THEN980:REM BOARD
150 IFLEFT$(ENTRY$,4)="OPEN"THEN1130:REM OPEN
160 IFLEFT$(ENTRY$,4)="DROP"THEN1300:REM DROP
170 IFLEFT$(ENTRY$,6)="UNLOCK"THEN860:REM UNLOCK
180 IFLEFT$(ENTRY$,4)="TYPE"THEN590:REM TYPE ON COMPUTER
190 IFLEFT$(ENTRY$,5)="SHOOT"THEN665:REM SHOOT
210 IFLEFT$(ENTRY$,5)="WINCH"THEN640:REM WINCH
220 IFLEFT$(ENTRY$,4)="FILL"THEN820 :REM EXAMINE
230 IFLEFT$(ENTRY$,4)="QUIT"THEN1800:REM QUIT
232 IFLEFT$(ENTRY$,7)="EXAMINE"THEN1500:REM EXAMINE
235 IFLEFT$(ENTRY$,3)="INV"THEN1600:REM INVENTORY
240 PRINTTAB(11)"[illegible]"
250 RETURN
260 REM TAKE
261 IFRIGHT$(ENTRY$,LEN(ENTRY$)-5)="LEAD"THENPRINT"YOU CAN'T":RETURN
262 IFLEN(ENTRY$)<5THENPRINT"VERB & NOUN, PLEASE":RETURN
265 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-5))="DYLTHIUM CRYSTALS"THENPRINT"YOU CAN'T":RET
URN
270 FORAA=0TOITEM(RN)
280 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-5))=ITEM$(RN,AA)THENBB=AA:AA=RN:GOTO300
285 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-5))=ITEM$(RN,AA)+"[illegible]"THENBB=AA:AA=RN:GOTO300
290 NEXT AA
```

```
294 PRINT"THERE ISN'T A ";RIGHT$(ENTRY$,(LEN(ENTRY$)-5));" HERE"
298 RETURN
300 INV$(INV)=ITEM$(RN,BB)
305 INV=INV+1:ITEM(RN)=ITEM(RN)-1
310 ITEM$(RN,BB)=""
320 RETURN
330 REM INSERT
335 IFLEN(ENTRY$)<7THENPRINT"VERB & NOUN, PLEASE":RETURN
340 IFRN=5ANDRIGHT$(ENTRY$,(LEN(ENTRY$)-7))="CIRCUIT"THENQQ=2:QQ$="CIRCUIT":GOTO
430
350 IFRN=5ANDRIGHT$(ENTRY$,(LEN(ENTRY$)-7))="ID"THENQQ=1:QQ$="ID":GOTO430
360 IFRN=6ANDRIGHT$(ENTRY$,(LEN(ENTRY$)-7))="CIRCUIT"ANDBO=1THEN500
370 IFRN=6ANDRIGHT$(ENTRY$,(LEN(ENTRY$)-7))="CIRCUIT"THENPRINT"WHERE?":RETURN
380 IFRN=8ANDRIGHT$(ENTRY$,(LEN(ENTRY$)-7))="ID"THEN520
390 IFRN=23ANDRIGHT$(ENTRY$,(LEN(ENTRY$)-7))="CRYSTAL"THEN560
410 IFRN=32ANDRIGHT$(ENTRY$,(LEN(ENTRY$)-7))="ID"THEN570
420 PRINT"YOU CAN'T INSERT A"RIGHT$(ENTRY$,(LEN(ENTRY$)-7)):RETURN
430 FORAA=0TOINV
431 IFINV$(AA)=QQ$THENBB=AA:AA=INV:GOTO435
433 NEXTAA
434 PRINT"YOU HAVEN'T GOT ONE":RETURN
435 OP=QQ
440 ONOPGOTO470,460
445 INV$(BB)=""
450 RETURN
460 IFOP=2THENPRINT"THE DOOR OPENS BUT THE CIRCUIT BURNS OUT":DO(RN)=1:RETURN
470 IFOP=1THENPRINT"THE DOOR OPENS SMOOTHLY":DO(RN)=1:RETURN
480 PRINT"THE CIRCUIT IS BROKEN"
490 RETURN
500 FORAA=0TOINV:IFINV$(AA)="CIRCUIT"THENBB=AA:AA=INV:GOTO504
502 NEXTAA
503 PRINT"YOU DON'T HAVE A CIRCUIT":RETURN
504 PRINT"THE ENGINE BURSTS INTO LIFE."
505 CI=1
507 FORAA=0TOINV:IFINV$(INV)="CIRCUIT"THENBB=AA:AA=INV:GOTO509
508 NEXT
509 INV$(BB)=""
510 RETURN
520 FORAA=0TOINV
530 IFINV$(AA)="ID"THENAA=INV:GOTO550
540 NEXTAA
545 PRINT"YOU DON'T HAVE AN ID.":RETURN
550 PRINT"THE LIFT DOOR OPENS.":LD=1:RETURN
560 FORAA=0TOINV:IFINV$(AA)="CRYSTAL"THENBB=AA:AA=INV:GOTO565
562 NEXTAA
563 PRINT"YOU DON'T HAVE A CRYSTAL":RETURN
565 PRINT"OKAY.":CO=1
566 FORAA=0TOINV:IFINV$(AA)="CRYSTAL"THENBB=AA:AA=INV:GOTO568
567 NEXTAA
568 INV$(BB)="":RETURN
570 FORAA=0TOINV:IFINV$(AA)="ID"THENBB=AA:AA=INV:GOTO579
572 NEXTAA
573 PRINT"YOU DON'T HAVE AN ID":RETURN
579 IFCO=1THENPRINT"THE COMPUTER COMES ON.":ID=1:RETURN
580 PRINT"OKAY.":RETURN
585 REM TYPE
590 IFRN=24THEN1400
595 IFLEN(ENTRY$)<5THENPRINT"VERB & NOUN, PLEASE":RETURN
600 IFRN=32ANDCO=1ANDID=1THEN620
605 IFRN=28THEN631
610 PRINT"ON WHAT?":RETURN
620 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-5))=CW$THENKY=1:RETURN
625 IFLEN(ENTRY$)<4THENPRINT"VERB & NOUN, PLEASE":GOTO5287
630 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-5))="COMBINATION"ANDKY=1THENC=2:RETURN
631 IFVAL(MID$(ENTRY$,5,(LEN(ENTRY$)-4)))=CBTHEN5830
632 RETURN
635 REM WINCH
640 IFRN=34THEN662
645 IFLEN(ENTRY$)<5THENPRINT"VERB & NOUN, PLEASE":RETURN
650 PRINT"YOU CAN'T.":RETURN
662 FORAA=0TOINV:IFINV$(AA)="WINCH"THENAA=INV:GOTO664
663 NEXTAA
664 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-6))="LEAD"THENINV=INV+1:INV$(INV)="LEAD":RETURN
665 REM SHOOT
667 IFLEN(ENTRY$)<5THENPRINT"VERB & NOUN, PLEASE":RETURN
670 IFRN=27THEN700
680 IFRN=39ORRN=40THEN740
690 PRINT"NOTHING HAPPENS.":RETURN
700 FORAA=0TOINV
710 IFINV$(AA)="BLASTER"THENAA=INV:GOTO730
720 NEXTAA
725 PRINT"YOU HAVEN'T GOT A BLASTER.":RETURN
730 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-6))="LOCK"THENPRINT"THE DRAW OPENS,";:GOTO732
731 GOTO740
732 ITEM(27)=ITEM(27)+1:ITEM$(27,ITEM)="KEY":DW=1
733 PRINT"AND YOU CAN SEE A KEY.":RETURN
740 FORAA=0TOINV
750 IFINV$(AA)="LASER CANNON"THENAA=INV:GOTO770
760 NEXTAA
765 PRINT"THE RAY BOUNCES OFF.":RETURN
770 FORAA=0TOINV
780 IFINV$(AA)="LEAD"THENAA=INV:GOTO800
790 NEXTAA
795 PRINT"THE ROCK, SHATTERED BY THE BLAST, KILLS":PRINT"YOU.":END
800 PRINT"AN OPENING IS REVILED":RF(RN)=1
810 RETURN
815 REM FILL
820 IFRN=34THEN840
825 IFLEN(ENTRY$)<5THENPRINT"VERB & NOUN, PLEASE":RETURN
830 PRINT"FROM WHAT":RETURN
840 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-5))="BOTTLE"THENBF=1:PRINT"OKAY.":RETURN
850 PRINT"HOW?":RETURN
855 REM UNLOCK
860 IFRN=33ANDRIGHT$(ENTRY$,(LEN(ENTRY$)-7))="CABINET"THEN880
865 IFLEN(ENTRY$)<7THENPRINT"VERB & NOUN, PLEASE":RETURN
870 PRINT"YOU CAN'T":RETURN
880 PRINT"YOU CAN SEE A PHOTON CANNON AND A LASER"
881 PRINT"CANNON."
882 ITEM(RN)=ITEM(RN)+1:ITEM$(RN,ITEM(RN))="PHOTON CANNON"
883 ITEM(RN)=ITEM(RN)+1:ITEM$(RN,ITEM(RN))="LASER CANNON"
884 RETURN
900 REM CONNECT
905 IFLEN(ENTRY$)<9THENPRINT"VERB & NOUN, PLEASE":RETURN
910 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-8))="WIRES"THEN930
920 PRINT"I CAN'T CONNECT THEM!!!":RETURN
930 IFRN=5THENOP=1:RETURN
940 IFRN=6THEN960
950 PRINT"THERE AREN'T ANY WIRES.":RETURN
960 PRINT"IN A BLAZING FLASH THE ENGINE EXPLODES.";:RETURN
970 REM BOARD
975 IFLEN(ENTRY$)<6THENPRINT"VERB & NOUN, PLEASE":RETURN
980 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-6))="MONORAIL"THEN1010
990 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-6))="LIFT"ANDLD=1THEN1020
1000 PRINT"YOU CAN'T BOARD THAT.":RETURN
1010 IFCI=1ANDRN=6THENPRINT"THE MONORAIL MOVES EAST INTO ANOTHER":PRINT"STATION"
1011 IFCI=1ANDRN=7THENPRINT"THE MONORAIL MOVES WEST INTO ANOTHER":PRINT"STATION"
1012 IFCI<>1THENPRINT"NOTHING HAPPENS.":RETURN
1015 FORXX=1TODY:NEXTXX:ONRN-5GOTO4000,3900
1020 IFRN<>8ANDRN<>22ANDRN<>29THENPRINT"THERE IS NO LIFT TO BOARD.":RETURN
1025 PRINT"DO YOU WISH TO GO UP OR DOWN";
1030 INPUT ENTRY$
1040 IFENTRY$="UP"THEN1070:REM ROOM ABOVE
1050 IFENTRY$="DOWN"THEN1100:REM ROOM BELOW
1060 PRINT"I SAID UP OR DOWN.";:GOTO1030
1070 IFRN=29THENPRINT"YOU CAN'T GO UP.":GOTO1030
1080 IFRN=8THEN6000:REM ROOM #29
1090 IFRN=22THEN4200:REM ROOM #8
1100 IFRN=22THENPRINT"YOU CAN'T GO DOWN.":GOTO1030
1110 IFRN=8THEN5000:REM ROOM #22
1120 IFRN=29THEN4200:REM ROOM #8
1125 REM OPEN
1130 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-5))="BONNET"THEN1150
1135 IFLEN(ENTRY$)<4THENPRINT"VERB & NOUN, PLEASE":RETURN
1140 PRINT"I CAN'T OPEN IT.":RETURN
1150 PRINT"YOU CAN SEE AN ENGINE WITH A CIRCUIT":PRINT"MISSING.":BO=1:RETURN
1292 IFLEN(ENTRY$)<5THENPRINT"VERB & NOUN, PLEASE":RETURN
```



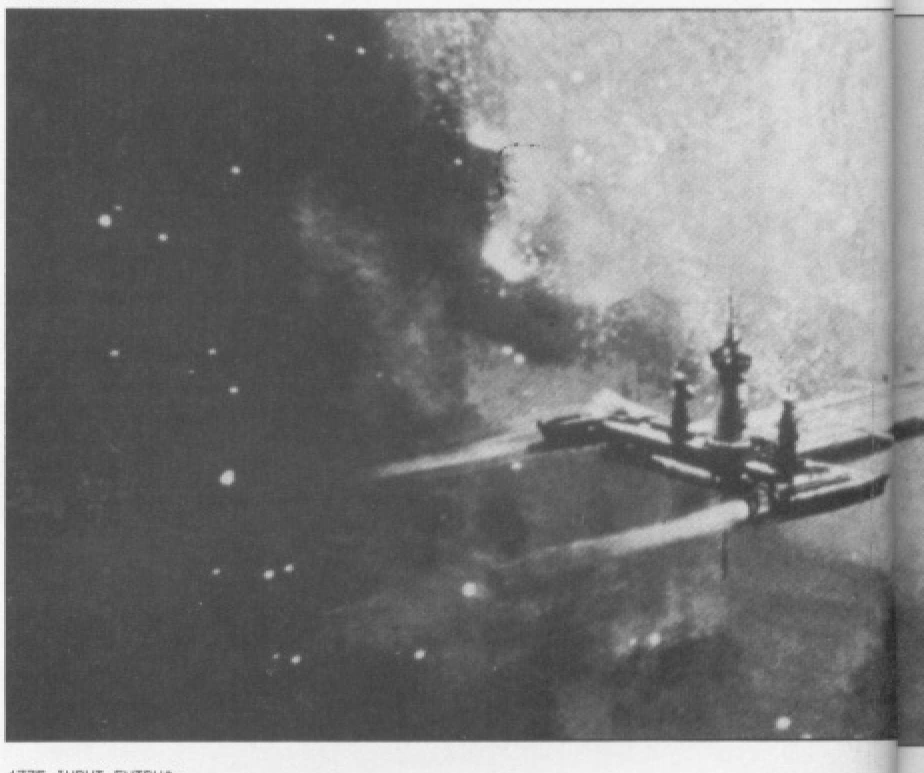
```
1295 REM DROP
1300 FORAA=0TOINV
1305 IFLEN(ENTRY$)<4THENPRINT"VERB & NOUN, PLEASE":RETURN
1310 IFINV$(AA)=RIGHT$(ENTRY$,(LEN(ENTRY$)-5))THENBB=AA:AA=INV:GOTO1330
1320 NEXTAA
1325 PRINT"YOU HAVEN'T GOT ONE TO DROP.":RETURN
1330 INV=INV-1:INV$(BB)=""
1340 ITEM(RN)=ITEM(RN)+1:ITEM$(RN,ITEM(RN))=RIGHT$(ENTRY$,(LEN(ENTRY$)-5))
1345 PRINT"OKAY."
1350 RETURN
1400 REM TYPE [ROOM #23]
1405 IFLEN(ENTRY$)<5THENPRINT"VERB & NOUN, PLEASE":RETURN
1410 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-5))=NN$THENPRINTCW$:RETURN
1420 PRINT"THAT IS INCORRECT.":RETURN
1500 REM EXAMINE
1505 IFLEN(ENTRY$)<7THENPRINT"VERB & NOUN, PLEASE":RETURN
1510 IFMID$(ENTRY$,8,(LEN(ENTRY$)-8))="OXYGEN TANK"THEN1530
1522 IFRIGHT$(ENTRY$,(LEN(ENTRY$)-8))="ID"THEN1700
1530 FORA=1TOITEM(RN)
1540 IFITEM$(RN,A)="OXYGEN TANK$"THENPRINT"IT'S EMPTY.":RETURN
1550 IFITEM$(RN,A)="OXYGEN TANK"THENPRINT"IT'S FULL.":RETURN
1560 NEXTA
1570 PRINT"THERE ISN'T A";MID$(ENTRY$,8,(LEN(ENTRY$)-8));"HERE."
```

```
1580 RETURN
1600 REM INVENTORY
1610 PRINT"INVENTORY":PRINT
1620 FORAA=0TOINV
1630 PRINTINV$(AA)
1640 NEXTAA
1650 PRINT""TAB(15)"PRESS A KEY"
1660 GETA$:IFA$=""THEN1660
1670 RETURN
1700 PRINT"""ID"
1710 PRINT"IT SAYS:";NN$
1720 PRINT""TAB(15)"PRESS A KEY"
1730 GETA$:IFA$=""THEN1730
1740 RETURN
1800 REM QUIT
1810 PRINT"DO YOU WANT ANOTHER GAME?"
1820 GETA$:IFA$=""THEN1820
1830 IFA$="Y"THENRUN
```



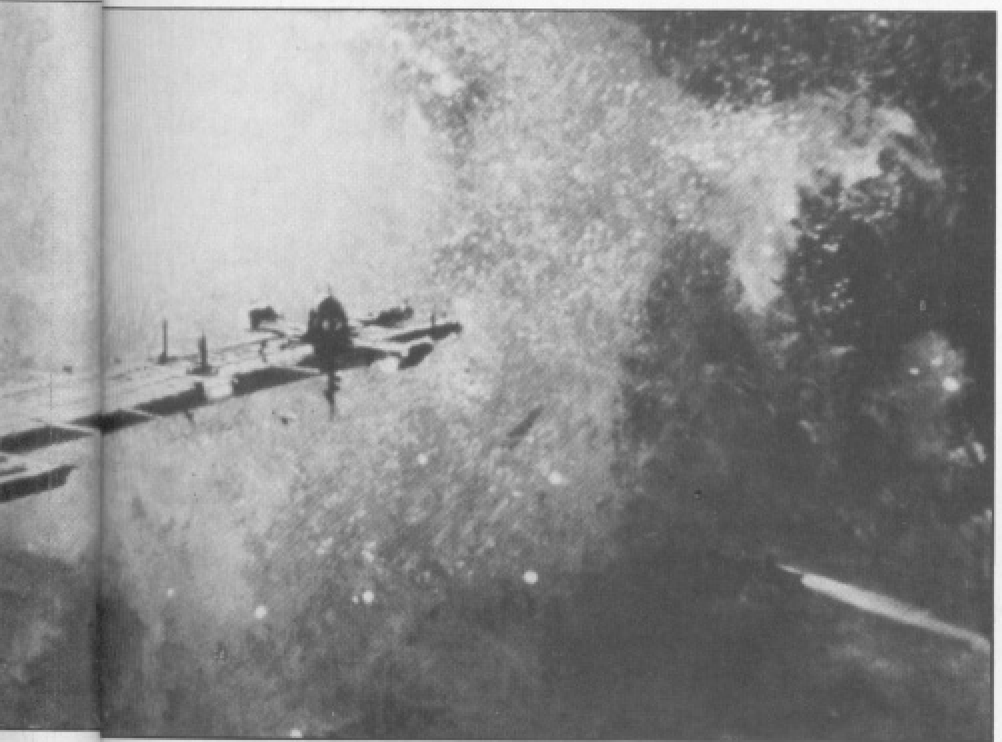
```
1840 IFA$="N"THENSTOP
1850 GOTO1820
2000 REM ********************
2010 REM *      ITEMS       *
2020 REM ********************
2025 DIM ITEM$(42,25)
2027 DIM ITEM(42):DIMRF(42)
2030 :
2040 RN=4
2050 ITEM(RN)=1:ITEM$(RN,1)="CIRCUIT"
2060 RN=5
2070 ITEM(RN)=1:ITEM$(RN,1)="ID"
2080 RN=6
2090 ITEM(RN)=1:ITEM$(RN,1)="TOOL BOX"
2100 FORRN=10TO17
2110 ITEM(RN)=1:ITEM$(RN,1)="SPACE SUIT"
2120 NEXTRN
2130 RN=20
2140 ITEM(RN)=1:ITEM$(RN,1)="BOTTLE"
2150 RN=21
2160 ITEM(RN)=1:ITEM$(RN,1)="CRYSTAL"
2170 RN=28
2180 ITEM(RN)=1:ITEM$(RN,1)="DYLITHIUM CRYSTALS"
2190 RN=41
2200 ITEM(RN)=1:ITEM$(RN,1)="WINCH"
2210 RN=39
2220 ITEM(RN)=1:ITEM$(RN,1)="OXYGEN TANK"
2230 RN=42
2240 ITEM(RN)=1:ITEM$(RN,1)="OXYGEN TANK"
2250 INV=0
2260 INV$(INV)="BLASTER"
2270 INV=INV+1
2280 RF(40)=0:RF(39)=0
2300 GOTO3400
2500 REM *** WINNING ROUTINE ***
2505 FORA=1TOINV
2506 IFINV$(A)="DYLITHIUM CRYSTALS"THENA=INV:GOTO2510
2507 NEXTA
2508 RETURN
2510 PRINT""
2511 POKE53265,PEEK(53265)AND191
2512 PRINT"          SIGNET"
2513 PRINT"                "
2514 PRINT"                "
2520 PRINT"";
2530 PRINT"YOU REPLACE THE CRYSTALS IN YOUR SHIP"
2540 PRINT"AND FLY OFF, BACK INTO THE MORE HOSPITA-";
2550 PRINT"BLE PARTS OF THE GALAXY."
2560 PRINT"";
2570 PRINTTAB(14)"WELL DONE!!!"
2575 PRINT"";
2580 GOTO1800
3000 REM *************
3010 REM *   ROOMS   *
3020 REM *************
3030 :
3040 REM ROOM #1
3045 RN=1
3050 PRINT"TECHNICAL WORKSHOP"
3060 PRINT"THIS IS A FAIRLY LARGE WORK ROOM DEDI-"
3070 PRINT"CATED TO THE MAINTENANCE OF THE SHUTTLES";
3080 PRINT"WHICH ONCE OPERATED FROM HERE. IN THE"
3090 PRINT"CENTRE IS A LARGE FROSTED GLASS WORK"
3100 PRINT"BENCH WITH VARIOUS TOOLS LAYING ON IT."
3101 PRINT"":PRINT
3102 FORAA=0TOITEM(RN)
3103 PRINTITEM$(RN,AA)
3104 NEXTAA
3110 PRINT"EXITS:"
3120 PRINT"SOUTH"
3130 INPUT ENTRY$
3140 IF ENTRY$="SOUTH"ORENTRY$="S"THEN3200
3150 GOSUB 100
3155 FORXX=1TODY:NEXTXX
3160 GOTO 3040
3200 REM ROOM #2
3205 RN=2
3210 PRINT"LANDING BAY #2"
3220 PRINT"THIS IS THE SECOND OF THE BASE'S LANDING";
3230 PRINT"AREAS INTHE CENTRE OF THE FLOORSPACE IS"
3240 PRINT"A LARGE DERILICT SHUTTLE THAT MUST HAVE"
3250 PRINT"LAID THERE FOR YEARS UNUSED. ALONG THE"
3260 PRINT"WALLS AND SPRAWLED OVER THE FLOOR ARE"
3270 PRINT"NUMEROUS TOOLS AND BITS OF SPACECRAFT"
3280 PRINT"MOST OF THEM USELESS."
3281 PRINT""
3282 FORAA=0TOITEM(RN)
3283 PRINTITEM$(RN,AA)
3284 NEXTAA
3290 PRINT"EXITS:"
3300 PRINT"EAST"
3310 PRINT"NORTH"
3320 INPUT ENTRY$
3330 IFENTRY$="EAST"ORENTRY$="E"THEN3600:ROOM #5
3340 IFENTRY$="NORTH"ORENTRY$="N"THEN3000
3350 GOSUB 100
3360 GOTO 3200
3365 FORXX=1TODY:NEXTXX
3400 REM ROOM #3
3405 RN=3
3406 GOSUB2500
3407 PRINT"LANDING BAY #1"
3410 PRINT"THIS IS THE FIRST OF THE BASES 2 LANDING";
3430 PRINT"AREAS. ALONG THE WALLS AND ACROSS THE"
3440 PRINT"FLOOR ARE VARIOUS TOOLS AND SPARES MOST"
3450 PRINT"ARE USELESS."
3451 PRINT"":PRINT
3452 FORAA=0TOITEM(RN)
3453 PRINTITEM$(RN,AA)
3454 NEXTAA
3460 PRINT"EXITS:"
3470 PRINT"EAST"
3480 PRINT"SOUTH"
3481 INPUT ENTRY$
3482 IFENTRY$="EAST"ORENTRY$="E"THEN3600
3483 IFENTRY$="SOUTH"ORENTRY$="S"THEN3500
3484 GOSUB 100
3485 FORXX=1TODY:NEXTXX
3486 GOTO 3400
3500 REM ROOM #4
3505 RN=4
3510 PRINT"TECHNICAL WORKSHOP"
3520 PRINT"THIS IS A FAIRLY LARGE WORK ROOM DEDI-"
3530 PRINT"CATED TO THE MAINTENANCE OF THE SHUTTLES";
3535 PRINT"WHICH ONCE OPERATED FROM HERE. IN THE"
3540 PRINT"CENTRE IS A LARGE FROSTED GLASS WORK"
3550 PRINT"BENCH WITH VARIOUS TOOLS LAYING ON IT."
3551 PRINT"":PRINT
3552 FORAA=0TOITEM(RN)
3553 PRINTITEM$(RN,AA)
3554 NEXTAA
3555 PRINT"EXITS:"
3560 PRINT"NORTH"
3565 INPUT ENTRY$
3570 IF ENTRY$="NORTH"ORENTRY$="N"THEN3400
3575 GOSUB 100
3577 FORXX=1TODY:NEXTXX
3580 GOTO3500
3600 REM ROOM #5
3605 RN=5
3610 PRINT"RECEPTION AREA"
3615 PRINT"THIS ROOM IS OBLONG WITH A LARGE ALUMIN-";
3620 PRINT"UIM AND WOOD DESK AT THE EASTERN END."
3625 PRINT"BEHIND THE DESK IS A LARGE BLOCK OF"
3630 PRINT"PIGEON HOLES WITH VARIOUS CARDS - MOST"
3635 PRINT"ARE EMPTY.NEXT TO THE DOOR IS A SLOT"
3670 PRINT"ABOVE WHICH IS AN OPEN CIRCUIT HOUSING"
3675 PRINT"WITH A CIRCUIT MISSING."
3680 PRINT""
3681 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
```

```
3882 PRINT"■■EXITS:■"
3883 PRINT"■NORTH WEST":PRINT"SOUTH WEST"
3884 IFDO(RN)=1THENPRINT"EAST"
3885 INPUT ENTRY$
3886 IFENTRY$="NORTH WEST"ORENTRY$="NW"THEN3200
3887 IFENTRY$="SOUTH WEST"ORENTRY$="SW"THEN3400
3888 IFENTRY$="EAST"ORENTRY$="E"ANDDO(RN)=1THEN3900
3889 GOSUB 100
3890 FORXX=1TODY:NEXTXX
3895 GOTO3600
3900 REM ROOM #6
3905 RN=6
3907 PRINT"■■MONORAIL STATION■"
3910 PRINT"■THIS IS A SMALL RECTANGULAR ROOM WITH A"
3915 PRINT"SINGULAR METAL TRACK RUNNING DOWN THE "
3920 PRINT"NORTHERN SIDE; STANDING ON THE TRACK IS"
3930 PRINT"A SMALL VEHICALE.IT APPEARS TO BE BROKEN";
3935 PRINT"■■Γ, →/ •‾‾_■"
3940 FORAA=1TOITEM(RN)
3945 PRINTITEM$(RN,AA)
3950 NEXTAA
3955 PRINT"■■EXITS:■"
3960 PRINT"■WEST"
3965 INPUT ENTRY$
3970 IFENTRY$="WEST"ORENTRY$="W"THEN3600
3980 GOSUB 100
3985 FORXX=1TODY:NEXTXX
3990 GOTO3900
4000 REM ROOM #7
4005 RN=7
4010 PRINT"■■MONORAIL STATION■"
4020 PRINT"■THIS IS BASICLY THE SAME AS THE OTHER"
4030 PRINT"STATION."
4040 PRINT"■■Γ, →/ •‾‾_■"
4045 FORAA=0TOITEM(RN)
4050 PRINTITEM$(RN,AA)
4060 NEXTAA
4070 PRINT"■■EXITS:■"
4080 PRINT"■EAST"
4090 INPUT ENTRY$
4100 IFENTRY$="EAST"ORENTRY$="E"THEN4200
4130 GOSUB 100
4135 FORXX=1TODY:NEXTXX
4140 GOTO4000
4200 REM ROOM #8
4205 RN=8
4210 PRINT"■■SECONDARY RECEPTION■".
4215 PRINT"■THIS IS THE SECOND RECEPTION AREA THAT"
4220 PRINT"ALLOWS ACCESS TO THE OTHER FLOORS THERE"
4225 PRINT"IS A DESK SIMILAR TO THAT IN THE FIRST"
4230 PRINT"RECEPTION AREA BUT SMALLER BESIDE WHICH"
4235 PRINT"IS STANDING AN EVELATOR WITH A SMALL RED";
4240 PRINT"SIGN FLASHING 'INSERT ID' ON IT; BEHIND"
4245 PRINT"THE DESK IS A DOOR."
4250 PRINT"■■Γ, →/ •‾‾_■":PRINT
4255 FORAA=0TOITEM(RN)
4260 PRINTITEM$(RN,AA)
4265 NEXTAA
4270 PRINT"■■EXITS:■"
4275 PRINT"NORTH EAST":PRINT"EAST":PRINT"SOUTH EAST"
4277 PRINT"WEST"
4280 INPUT ENTRY$
4285 IFLEFT$(ENTRY$,1)="W"THEN4000
4290 IFENTRY$="NORTH EAST"ORENTRY$="NE"THEN4800:REM ROOM #20
4291 IFENTRY$="EAST"ORENTRY$="E"THEN4300
4292 IFENTRY$="SOUTH EAST"ORENTRY$="SE"THEN4900:REM ROOM #21
4293 GOSUB 100
4294 FORXX=1TODY:NEXTXX
4295 GOTO4200
4300 REM ROOM #9
4305 RN=9
4310 PRINT"■■CORRIDOR■"
4315 PRINT"■THIS IS THE CENTRAL CORRIDOR OF THIS" *
4320 PRINT"FLOOR LEADING TO ALL THE STAFF QUARTERS,";
4325 PRINT"WASH ROOM ETC. IT IS FAIRLY LONG AND IS"
4330 PRINT"ONLY USED FOR ACCESS."
4335 PRINT"■■Γ, →/ •‾‾_■":PRINT
4340 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
4345 PRINT"■■EXITS:■":PRINT
4350 PRINT"NORTH WEST":PRINT"NORTH NORTH WEST":PRINT"NORTH NORTH EAST":PRINT"NOR
TH EAST"
4355 PRINT"EAST":PRINT"SOUTH EAST":PRINT"SOUTH SOUTH EAST":PRINT"SOUTH SOUTH WES
T"
4357 PRINT"SOUTH WEST"
4360 PRINT"WEST"
4365 INPUT ENTRY$
4370 IFENTRY$="WEST"ORENTRY$="W"THEN4200
4371 IFENTRY$="EAST"ORENTRY$="E"THEN4600:REM ROOM #18
4372 IFENTRY$="NORTH WEST"ORENTRY$="NW"ORENTRY$="NORTH EAST"ORENTRY$="NE"THEN440
0
4373 IFENTRY$="NORTH NORTH EAST"ORENTRY$="NNW"THEN4400
4374 IFENTRY$="NORTH NORTH EAST"ORENTRY$="NNE"THEN4400
4375 IFENTRY$="SOUTH WEST"ORENTRY$="SW"ORENTRY$="SOUTH EAST"ORENTRY$="SE"THEN440
0
4376 IFENTRY$="SOUTH SOUTH EAST"ORENTRY$="SSW"THEN4400
4377 IFENTRY$="SOUTH SOUTH EAST"ORENTRY$="SSE"THEN4400
4380 GOSUB 100
4385 FORXX=1TODY:NEXTXX
4390 GOTO4300
4400 REM ROOMS 10,11,12 & 13
4410 PRINT"■■STAFF QUARTERS■"
4415 RN=10
4420 PRINT"■THIS IS A STANDARD BEDROOM/LIVING ROOM"
4430 PRINT"AREA; THERE IS A BED ALONG THE EASTERN"
4440 PRINT"WALL BEYOND WHCH ARE SOME ARMCHAIRS AND"
4450 PRINT"A VIDEO SCREEN TO THE WEST IS A LARGE "
4455 PRINT"CUPBOARD CONTAINING STANDARD ISSUE DRESS";
4460 PRINT"■■Γ, →/ •‾‾_■"
4465 FORAA=0TOITEM(RN)
4470 PRINTITEM$(RN,AA)
4475 NEXTAA
4480 PRINT"■■EXITS:■"
4485 PRINT"■SOUTH"
4490 INPUT ENTRY$
4491 IFENTRY$="SOUTH"ORENTRY$="S"THEN4300
4492 GOSUB100
4494 FORXX=1TODY:NEXTXX
4495 GOTO4400
4500 REM ROOM #14,15,16 & 17
4510 PRINT"■■STAFF QUARTERS■"
4515 RN=14
4520 PRINT"■THIS IS A STANDARD BEDROOM/LIVING ROOM"
4530 PRINT"AREA; THERE IS A BED ALONG THE EASTERN"
4540 PRINT"WALL BEYOND WHCH ARE SOME ARMCHAIRS AND"
4550 PRINT"A VIDEO SCREEN TO THE WEST IS A LARGE "
4555 PRINT"CUPBOARD CONTAINING STANDARD ISSUE DRESS";
4560 PRINT"■■Γ, →/ •‾‾_■"
4565 FORAA=0TOITEM(RN)
4570 PRINTITEM$(RN,AA)
4575 NEXTAA
4580 PRINT"■■EXITS:■"
4585 PRINT"■SOUTH"
4590 INPUT ENTRY$
4591 IFENTRY$="SOUTH"ORENTRY$="S"THEN4300
4592 GOSUB100
4593 GOTO4500
4594 FORXX=1TODY:NEXTXX
4595 GOTO4500
4600 REM ROOM #18
4605 RN=18
4610 PRINT"■■GYM■"
4620 PRINT"■THIS IS A WORKOUT AREA AND SEEMS THE"
4625 PRINT"ONLY RECREATION FACILITY,EXCEPT THE BARS";
4630 PRINT"ON THE BASE. IT CONTAINS ALL THE NEEDED"
4635 PRINT"EQUIPMENT THOUGH MUCH OF IT IS NOW IN"
4640 PRINT"DISREPAIR."
4645 PRINT"■■Γ, →/ •‾‾_■"
4650 FORAA=0TOITEM(RN):PRINT
4655 PRINTITEM$(RN,AA)
4660 NEXTAA
4665 PRINT"■■EXITS:":PRINT
4670 PRINT"EAST":PRINT"WEST"
4675 INPUT ENTRY$
4680 IFENTRY$="EAST"ORENTRY$="E"THEN4700
4681 IFENTRY$="WEST"ORENTRY$="W"THEN4300
4690 GOSUB100
4692 FORXX=1TODY:NEXTXX
4695 GOTO4600
4700 REM ROOM #19
4710 PRINT"■■WASH ROOM■"
4715 RN=19
4720 PRINT"■THIS SMALL ROOM CONTAINS A SHOWER,A BATH";
4730 PRINT"SEVERAL SINKS AND A WC."
4740 PRINT"■■Γ, →/ •‾‾_■":PRINT
4750 FORAA=1TOITEM(RN)
4755 PRINTITEM$(RN,AA)
4760 NEXTAA
4765 PRINT"■■EXITS:■":PRINT
4770 PRINT"WEST"
```



```
4775 INPUT ENTRY$
4780 IFENTRY$="WEST"ORENTRY$="W"THEN4600
4785 IFENTRY$="PEE"THENPRINT"YOU RELIEVE YOURSELF.":GOTO4700
4787 FORXX=1TODY:NEXTXX
4790 GOTO4700
4800 REM ROOM #20
4805 RN=20
4810 PRINT"■■BAR■"
4825 PRINT"■AN OAK BAR COVERS ONE WALL AROUND WHICH";
4830 PRINT"ARE TABLES AND CHAIRS."
4840 PRINT"■■Γ, →/ •‾‾_■":PRINT
4850 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
4860 PRINT"■■EXITS:■"
4870 PRINT"■SOUTH WEST":PRINT
4875 INPUT ENTRY$
4880 IFENTRY$="SOUTH WEST"ORENTRY$="SW"THEN4200
4885 GOSUB100
4887 FORXX=1TODY:NEXTXX
4890 GOTO4800
4900 REM ROOM #21
4905 RN=21
4910 PRINT"■■BAR #2 [TEMPLE]"
4915 PRINT"■BEFORE IT BECAME A TEMPLE THIS USED TO"
4920 PRINT"BE A BAR AND STILL HAS THE BAR ALONG THE";
4925 PRINT"SOUTHERN WALL.THIS HAS NOW HOWEVER BEEN"
4930 PRINT"DRAPED WITH A RED CLOTH AND TWO CANDLES"
4935 PRINT"STAND AT EACH END.AN INSCRIPTION HAS"
4940 PRINT"BEEN CARVED INTO THE FRONT IN DIGITS YOU";
4945 PRINT"RECOGNISE TO BE OF ANOTHER CULTURE.";
4946 FORA=1TOITEM(RN):IFITEM$(RN,A)="CRYSTAL"THENA=ITEM(RN):GOTO4950
4947 NEXTA
4948 GOTO4975
4950 PRINT"IN   THE MIDDLE OF THE ALTER IS A TALL"
4955 PRINT"CYNDRICAL CRYSTAL,SIMILAR IN SIZE AND"
4960 PRINT"SHAPE TO YOUR LASER CARTRIGE ALTHOUGH IT";
4965 PRINT"IS NOT THE FIMILAR BLUE BUT A RED AND"
4970 PRINT"WHITE FROSTED EFECT."
4975 PRINT:PRINT"■Γ, →/ •‾‾_■":PRINT
4977 CY=2
```

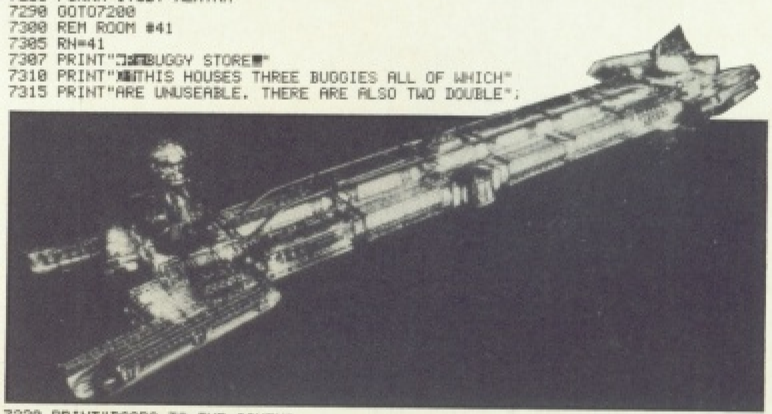
```
4980 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):IFITEM$(RN,AA)="CRYSTAL"THENCY=1
4981 NEXTAA
4982 PRINT"■■EXITS:■"
4983 PRINT"■■NORTH WEST"
4984 INPUT ENTRY$
4985 IFENTRY$="NORTH WEST"ORENTRY$="NW"THEN4200
4990 GOSUB100
4992 FORXX=1TODY:NEXTXX
4995 GOTO4900
5000 REM LOWER LEVEL
5010 REM ROOM #22
5015 RN=22
5020 PRINT"□■LOWER LEVEL RECEPTION AREA■"
5030 PRINT"■■THIS ROOM IS INTRISICLY THE SAME AS THE"
5040 PRINT"OTHER RECEPTION AREAS IN THE COMPLEX"
5050 PRINT"EXCEPT FOR SOME MINOR DETAILS."
5055 PRINT"■■ ╔, ─♦/ ●‾‾_■":PRINT
5060 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
5065 PRINT"■■EXITS:■"
5070 PRINT"■■EAST":PRINT"NORTH EAST"
5075 INPUT ENTRY$
5080 IFLEFT$(ENTRY$,1)="E"THEN5100
5085 IFENTRY$="NORTH EAST"ORENTRY$="NE"THEN5200:REM ROOM #24
5090 GOSUB 100
5092 FORXX=1TODY:NEXTXX
5095 GOTO5010
5100 REM ROOM #23
5105 RN=23
5110 PRINT"□■COMPUTER MEMORY BANKS■"
5115 PRINT"■■THIS HOUSES ALL THE MEMORY FOR THE MAIN"
5120 PRINT"COMPUTER SYSTEM WHICH IS NOW";
5122 IFCO=1THENPRINT" ON ";:GOTO5130
5125 PRINT" SHUT DOWN"
5130 PRINT"THE ROOM IS RECTANGULAR WITH LARGE"
5135 PRINT"METAL CABINETS RUNNING DOWN THE SOUTHERN";
5140 PRINT"AND NORTHERN WALLS.IN THE MIDDLE OF THE"
5145 PRINT"SOUTHERN WALL IS A POST WITH A LARGE RED";
5150 PRINT"SWITCH CASING ON IT.IN THE CENTRE IS A"
5155 PRINT"CIRCULAR HOLE WITH THE WORDS:'INSERT'"
5160 PRINT"WRITTEN BELOW IT ON A METAL PLAQUE."
5165 PRINT"■■ ╔, ─♦/ ●‾‾_■":PRINT
```



```
5170 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
5180 PRINT"■■EXITS:■"
5182 PRINT"■■SOUTH WEST":PRINT"SOUTH EAST"
5185 INPUT ENTRY$
5187 IFENTRY$="SOUTH WEST"ORENTRY$="SW"THEN5000
5190 IFENTRY$="SOUTH EAST"ORENTRY$="SE"THEN5400
5192 GOSUB100
5193 FORXX=1TODY:NEXTXX
5195 GOTO5100
5200 REM ROOM #24
5205 RN=24
5210 PRINT"□■RECORDS ROOM■"
5215 PRINT"■■THIS ROOM IS SIMILAR TO THE COMPUTER"
5220 PRINT"BANKS WITH THE SAME ALUMINUIM CABINETS"
5225 PRINT"LINING IT,BUT INSTEAD OF A POST THERE IS";
5230 PRINT"A COMPUTER TERMINAL. THIS IS DISPLAYD:"
5235 PRINT"■■"TAB(5)"/// RECORDS - CLASSIFIED ///"
5237 PRINT"■■ENTER CARD NUMBER AND NAME >"
5240 PRINT"■■ ╔, ─♦/ ●‾‾_■":PRINT
5245 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
5250 PRINT"■■EXITS:■"
5255 PRINT"■■EAST":PRINT"WEST"
5260 INPUT ENTRY$
5265 IFENTRY$="EAST"ORENTRY$="E"THEN5400
5270 IFENTRY$="WEST"ORENTRY$="W"THEN5000
5280 GOSUB100
5285 FORXX=1TODY:NEXTXX
5287 GETA$:IFA$=""THEN5287
5290 GOTO5200
5400 REM ROOM #25
5405 RN=25
5410 PRINT"□■SECURITY POST■"
5415 PRINT"■■THIS ROOM IS SPARSLEY FURNISHED WITH"
5420 PRINT"ONLY THE MINIMUM OF COMFORT."
5425 PRINT"■■ ╔, ─♦/ ●‾‾_■":PRINT
5430 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
5435 PRINT"■■EXITS:"
5440 PRINT"■■SOUTH WEST":PRINT"EAST":PRINT"WEST"
5442 INPUT ENTRY$
5445 IFENTRY$="SOUTH WEST"ORENTRY$="SW"THEN5200
5450 IFENTRY$="EAST"ORENTRY$="E"THEN5500
5455 IFENTRY$="WEST"ORENTRY$="W"THEN5100
5460 GOSUB100
5465 FORXX=1TODY:NEXTXX
5470 GOTO5400
5500 REM ROOM #26
5505 RN=26
5510 PRINT"□■LIFE SUPPORT■"
5515 PRINT"■■THIS ROOM HOUSES THE LIFE SSUPPORT"
5520 PRINT"SYSTEM FOR THE WHOLE COMPLEX."
5525 PRINT"■■ ╔, ─♦/ ●‾‾_■"
5530 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
5540 PRINT"■■EXITS:■"
5550 PRINT"■■WEST":PRINT"EAST"
5560 INPUT ENTRY$
5565 IFENTRY$="WEST"ORENTRY$="W"THEN5400
5570 IFENTRY$="EAST"ORENTRY$="E"THEN5600
5580 GOSUB100
5585 FORXX=1TODY:NEXTXX
5590 GOTO5500
5600 REM ROOM #27
5605 RN=27
5610 PRINT"□■CENTRAL SECURITY■"
5620 PRINT"■■THIS IS ANOTHER SECURITY POST, THE MAIN";
5630 PRINT"ONE FOR THE BASE.THERE ARE SCATTERED"
5635 PRINT"CHAIRS AND DESKS AND A LARGE FILLING"
5640 PRINT"CABINET.";
5645 IFDW=1THENPRINT"THE DRAW IS OPEN.":GOTO5655
5650 PRINT"THE DRAW IS LOCKED."
5655 PRINT"■■ ╔, ─♦/ ●‾‾_■":PRINT
5660 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
5670 PRINT"■■EXITS:■"
5675 PRINT"■■EAST":PRINT"WEST"
5680 INPUT ENTRY$
5682 IFENTRY$="EAST"ORENTRY$="E"THEN5700
5685 IFENTRY$="WEST"ORENTRY$="W"THEN5500
5690 GOSUB100
5692 FORXX=1TODY:NEXTXX
5695 GOTO5600
5700 REM ROOM #28
5705 RN=28
5710 PRINT"□■CENTRAL CORE■"
5720 PRINT"■■THIS ROOM IS FAIRLY EMPTY EXCEPT FOR A"
5725 PRINT"CABINET AT ONE END IN WHICH THE DYLITHIUM";
5730 PRINT"CRYSTALS ARE HOUSED,IN WATER,LOCKED"
5735 PRINT"DOWN WITH A BAR CONTROLED BY A COMBIN-"
5740 PRINT"ATION LOCK."
5750 PRINT"■■ ╔, ─♦/ ●‾‾_■":PRINT
5755 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
5760 PRINT"■■EXITS:■"
5765 PRINT"■■WEST"
5770 INPUT ENTRY$
5775 IFENTRY$="WEST"ORENTRY$="W"THEN5600
5776 FORA=1TOINV:IFINV$(A)="DYLITHIUM CRYSTALS"THENA=INV:GOTO5780
5777 NEXTA
5778 IFC9=1THEN5845
5780 GOSUB100
5785 FORXX=1TODY:NEXTXX
5790 GOTO5700
5830 PRINT"THE CABINET OPENS"
5840 C9=1:GOTO5770
5845 IFLEFT$(ENTRY$,3)="PUT"ANDRIGHT$(ENTRY$,6)="BOTTLE"THEN5860
5850 IFLEFT$(ENTRY$,5)="PLACE"ANDRIGHT$(ENTRY$,6)="BOTTLE"THENWL=5:GOTO5860
5855 GOTO5870
5860 FORAA=0TOINV:IFINV$(AA)="BOTTLE"THENAA=INV:GOTO5865
5861 NEXTAA
5862 PRINT"YOU DON'T HAVE A BOTTLE.":GOTO5870
5865 IFBF=1THEN5900
5870 PRINT"THE CRYSTALS SHRIVAL UP AND BECOME USE-":PRINT"LESS.":END
5900 PRINT"OKAY."
5901 FORAA=0TOINV:IFINV$(AA)="SPACE SUIT"THENAA=INV:GOTO5904
5902 NEXTAA
5903 PRINT"YOU HAVE NO SPACE SUIT YOU DIE OF ATMOS-";:PRINT"PHERE POISONING.":EN
D
5904 FORAA=0TOINV:IFINV$(AA)="OXYGEN TANK"THENAA=INV:GOTO5910
5905 NEXTAA
5906 PRINT"YOU HAVE NO OXYGEN YOU DIE OF ATMOSPHERE";:PRINT"POISONING.":END
5909 PRINT"OKAY."
5910 FORA=0TOITEM(RN)
5920 IFITEM$(RN,A)="DYLITHIUM CRYSTALS"THENITEM$(RN,A)="":A=ITEM(RN):GOTO5940
5930 NEXTA
5940 INV=INV+1:INV$(INV)="DYLITHIUM CRYSTALS"
5950 GOTO5700
6000 REM UPPER FLOOR
6010 REM ROOM #29
6015 RN=29
6020 PRINT"□■UPPER RECEPTION AREA■"
6025 PRINT"■■THIS IS BASICLY THE SAME AS THE SECOND-"
6030 PRINT"ARY RECEPTION AREA."
6035 PRINT"■■ ╔, ─♦/ ●‾‾_■":PRINT
6040 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
6045 PRINT"■■EXITS:■"
6050 PRINT"■■EAST"
6055 INPUT ENTRY$
6060 IFLEFT$(ENTRY$,1)="E"THEN6100
6070 GOSUB100
6075 FORXX=1TODY:NEXTXX
6080 GOTO6010
6100 REM ROOM #30
6105 RN=30
6110 PRINT"□■SECURITY POST■"
6115 PRINT"■■THIS POST IS ONLY SPARSLY FURNISHED WITH";
6120 PRINT"ONLY THE MINIMUM OF COMFORT."
6125 PRINT"■■ ╔, ─♦/ ●‾‾_■":PRINT
6130 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
6135 PRINT"■■EXITS:■"
6140 PRINT"■■EAST":PRINT"WEST"
6145 INPUT ENTRY$
6150 IFLEFT$(ENTRY$,1)="E"THEN6200
6160 IFLEFT$(ENTRY$,1)="W"THEN6000
6170 GOSUB100
6175 FORXX=1TODY:NEXTXX
6180 GOTO6100
6200 REM ROOM #31
6205 RN=31
6210 PRINT"□■CENTRAL CONTROL ROOM■"
6215 PRINT"■■THIS ROOM CONTROLS EVERYTHING IN THE"
6220 PRINT"BASE; THIS IS EASILY OBVIOUS BY THE"
6225 PRINT"BANKS OF INSTRUMENTS AND ENDLESS SUPLIES";
6230 PRINT"OF CONSOLES. SOME OF THEM ARE ON BUT A"
6235 PRINT"MAJORITY ARE BROKEN PROBABLY UNUSED FOR"
6240 PRINT"DECADES."
6245 PRINT"■■ ╔, ─♦/ ●‾‾_■":PRINT
6250 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
6255 PRINT"■■EXITS:■"
6260 PRINT"■■NORTH":PRINT"EAST"
```

Program listing

```
6265 PRINT"SOUTH":PRINT"WEST"
6270 INPUT ENTRY$
6275 IFLEFT$(ENTRY$,1)="N"THEN6300
6277 IFLEFT$(ENTRY$,1)="E"THEN6600:REM ROOM #34
6280 IFLEFT$(ENTRY$,1)="S"THEN6500:REM ROOM #33
6285 IFLEFT$(ENTRY$,1)="W"THEN6100:REM ROOM #30
6290 GOSUB100
6292 FORXX=1TODY:NEXTXX
6295 GOTO6200
6300 REM ROOM #32
6305 RN=32
6310 PRINT"██COMPUTER ROOM█"
6320 PRINT"FOR THE WHOLE COMPLEX.IT IS A SMALL UNIT";
6325 PRINT"ON THE NORTHERN WALL OF THE ROOM.IT HAS"
6330 PRINT"A MONITOR AND KEYBOARD.IT APPEARS TO BE"
6335 PRINT"OFF.";
6340 IFCO=1THENPRINT"A SMALL RED PANEL FLASHES:":PRINT"██▬,/•¯_|  ,¯█"
6345 IFID=1THEN6400
6350 PRINT:PRINT"██ Γ, ¬+/ •¯¯_█"
6355 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
6360 PRINT"██EXITS:█"
6365 PRINT"██SOUTH"
6367 IFKY=1THENPRINTCB
6370 INPUT ENTRY$
6375 IFLEFT$(ENTRY$,1)="S"THEN6200
6380 GOSUB100
6385 FORXX=1TODY:NEXTXX
6390 GOTO6300
6400 REM
6405 PRINT"█"TAB(8)"█///CENTRAL COMPUTER///"
6410 PRINT"██ PLEASE TYPE ENTRY KEYWORD"
6420 GOTO6350
6430 PRINTCW$:PRINT" _¯+¯ |"
6435 IFKY=1THENPRINTCB
6440 GOTO6350
6500 REM ROOM #33
6505 RN=33
6510 PRINT"██ARMOURY█"
6515 PRINT"██THIS IS A SMALLISH ROOM WHICH CONTAINS"
6520 PRINT"THE MEAGER WEAPONRY FOR THE BASE. ON"
6525 PRINT"SOUTHERN WALL IS A LARGE PLATE CABINET"
6530 PRINT"CONTAINING THE WEAPONS IT NEEDS A KEY TO";
6535 PRINT"BE OPENED."
6540 PRINT"██ Γ, ¬+/ •¯¯_█":PRINT
6545 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
6550 PRINT"██EXITS:█"
6555 PRINT"██NORTH":PRINT"SOUTH"
6560 INPUT ENTRY$
6565 IFLEFT$(ENTRY$,1)="N"THEN6200
6570 IFLEFT$(ENTRY$,1)="S"ANDMID$(EN$,2,1)<>"H"THEN7300:REM ROOM #41
6580 GOSUB100
6585 FORXX=1TODY:NEXTXX
6590 GOTO6500
6600 REM ROOM #34
6605 RN=34
6610 PRINT"██LAKE█"
6615 PRINT"██THIS IS THE AGRICULTRAL CENTRE OF THE"
6620 PRINT"BASE - YOU ARE STANDING BY A FAIR SIZED"
6625 PRINT"LAKE WHICH HAS ITS SUPPLY FROM A STREAM"
6630 PRINT"RUNNING DOWN FROM SOME HILLS TO THE"
6635 PRINT"NORTH.THOUGH THE SKYLIGHT ABOVE COMES "
6640 PRINT"SILVERY GLANCE OF THE SINGLE MOON.AT"
6645 PRINT"THE BOTTOM OF THE LAKE YOU CAN SEE A"
6647 PRINT"SHEET OF BLACK METAL"
6650 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
6655 PRINT"██EXITS:█"
6660 PRINT"██WEST":PRINT"EAST":PRINT"NORTH EAST"
6665 INPUT ENTRY$
6670 IFENTRY$="WEST"ORENTRY$="W"THEN6200
6675 IFLEFT$(ENTRY$,1)="E"THEN6800:REM ROOM #36
6677 IFENTRY$="NORTH EAST"ORENTRY$="NE"THEN6700:REM ROOM #35
6680 GOSUB100
6685 FORXX=1TODY:NEXTXX
6690 GOTO6600
6700 REM ROOM #35
6705 RN=35
6710 PRINT"██WATERFALL█"
6715 PRINT"██THIS IS A WATERFALL THAT SUPPLIES THE"
6720 PRINT"LAKE TO THE SOUTH WEST IT IS BEAUTIFUL"
6725 PRINT"BUT HAD NO REAL PURPOSE."
6730 PRINT"██ Γ, ¬+/ •¯¯_█":PRINT
6735 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
6740 PRINT"██EXITS:█"
6745 PRINT"██SOUTH WEST":PRINT"EAST"
6750 INPUT ENTRY$
6755 IFENTRY$="SOUTH WEST"ORENTRY$="SW"THEN6600
6760 IFLEFT$(ENTRY$,1)="E"THEN6900:REM ROOM #37
6765 GOSUB100
6767 FORXX=1TODY:NEXTXX
6780 GOTO6700
6800 REM ROOM #36
6805 RN=36
6810 PRINT"██CROPS█"
6815 PRINT"██THIS LARGE EXPANSE OF DESOLATE LAND USED";
6820 PRINT"TO BE USED TO GROW CROPS."
6825 PRINT"██ Γ, ¬+/ •¯¯_█":PRINT
6830 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
6835 PRINT"██EXITS:█"
6840 PRINT"██NORTH EAST":PRINT"WEST"
6850 INPUT ENTRY$
6855 IFENTRY$="NORTH EAST"ORENTRY$="NE"THEN6900
6860 IFLEFT$(ENTRY$,1)="W"THEN6700
6870 GOSUB100
6875 FORXX=1TODY:NEXTXX
6880 GOTO6800
6900 REM ROOM #37
6905 RN=37
6910 PRINT"██ENTRANCE CAVE█"
6915 PRINT"██THIS IS THE FIRST OF A COMPLEX OF CAVES"
6920 PRINT"THAT ULTIMATELY LEADS TO THE GENERAL "
6930 PRINT"██ Γ, ¬+/ •¯¯_█"
6935 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
6940 PRINT"██EXITS:█"
6945 PRINT"██SOUTH WEST":PRINT"WEST":PRINT"SOUTH":PRINT"EAST"
6950 INPUT ENTRY$
6960 IFENTRY$="SOUTH WEST"ORENTRY$="SW"THEN6800
6965 IFLEFT$(ENTRY$,1)="W"THEN6700
6970 IFLEFT$(ENTRY$,1)="S"THEN7000
6975 IFLEFT$(ENTRY$,1)="E"THEN7100:REM ROOM #39
6980 GOSUB100
6985 FORXX=1TODY:NEXTXX
6990 GOTO6900
7000 REM ROOM #38
7005 RN=38
7010 PRINT"██LAIR█"
7025 PRINT"██THIS CAVE IS LARGER THEN THE FIRST AND"
7030 PRINT"BY THE STRAW AND GENERAL MESS IT IS"
7035 PRINT"OBVIOUS THAT THIS IS A LAIR FOR SOME"
7040 PRINT"CREATURE."
7045 PRINT"██ Γ, ¬+/ •¯¯_█":PRINT
7050 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
7055 PRINT"██EXITS:█"
7060 PRINT"██NORTH":PRINT"EAST"
7065 INPUT ENTRY$
7070 IFLEFT$(ENTRY$,1)="N"THEN6900
7075 IFLEFT$(ENTRY$,1)="E"THEN7200:REM ROOM #40
7080 GOSUB100
7085 FORXX=1TODY:NEXTXX
7090 GOTO7000
7100 REM ROOM #39
7105 RN=39
7110 PRINT"██CAVERN█"
7115 PRINT"██THIS SMALL CAVE ONCE FORMED THE ENTER-"
7120 PRINT"ANCE TO THE GENERAL STORE."
7125 IFRF(RN)=0THENPRINT"BUT IS NOW BLOCKED BY A ROCK FALL."
7130 PRINT"██ Γ, ¬+/ •¯¯_█":PRINT
7135 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
7140 PRINT"██EXITS:█"
7145 PRINT"██WEST"
7150 IFRF(RN)=1THENPRINT"EAST"
7155 INPUT ENTRY$
7160 IFLEFT$(ENTRY$,1)="W"THEN6900
7165 IFLEFT$(ENTRY$,1)="E"ANDRF(RN)=1THEN7500:REM ROOM #42
7167 IFENTRY$="TAKE OXYGEN TANK"THENENTRY$=ENTRY$+"█"
7170 GOSUB100
7175 FORXX=1TODY:NEXTXX
7180 GOTO7100
7200 REM ROOM #40
7210 RN=40
7220 PRINT"██CAVE█"
7225 PRINT"██THIS IS A SMALL CAVE OF NO IMPORTANCE"
7230 IFRF(RN)=0THENPRINT"THERE IS A ROCK FALL TO THE EAST"
7240 PRINT"██ Γ, ¬+/ •¯¯_█":PRINT
7245 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
7250 PRINT"██EXITS:█"
7255 PRINT"██WEST"
7260 IFRF(RN)=1THENPRINT"EAST"
7265 INPUT ENTRY$
7270 IFLEFT$(ENTRY$,1)="W"THEN7000
7275 IFLEFT$(ENTRY$,1)="E"ANDRF(RN)=1THEN7500:REM ROOM #42
7280 GOSUB100
7285 FORXX=1TODY:NEXTXX
7290 GOTO7200
7300 REM ROOM #41
7305 RN=41
7307 PRINT"██BUGGY STORE█"
7310 PRINT"██THIS HOUSES THREE BUGGIES ALL OF WHICH"
7315 PRINT"ARE UNUSEABLE. THERE ARE ALSO TWO DOUBLE";
```



```
7320 PRINT"DOORS TO THE SOUTH"
7325 PRINT"██ Γ, ¬+/ •¯¯_█":PRINT
7330 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
7335 PRINT"██EXITS:█"
7340 PRINT"██NORTH":PRINT"SOUTH
7350 INPUT ENTRY$
7355 IFLEFT$(ENTRY$,1)="N"THEN6500:REM ROOM #33
7360 IFLEFT$(ENTRY$,1)="S"THEN7400
7370 GOSUB100
7375 FORXX=1TODY:NEXTXX
7380 GOTO7300
7400 PRINT"██PLANET SURFACE█"
7410 PRINT"██YOU TRAVEL FOR ABOUT TWO HOURS BEFORE"
7415 PRINT"FROM ATMOSPHERE POISONING."
7420 END
7500 REM ROOM #42
7505 RN=42
7510 PRINT"██GENERAL STORE█"
7515 PRINT"██THIS CONTAINS ALL THE MAJOR SUPPLIES FOR";
7520 PRINT"THE BASE, MOST OF IT IS JUNK."
7530 PRINT"██ Γ, ¬+/ •¯¯_█":PRINT
7535 FORAA=0TOITEM(RN):PRINTITEM$(RN,AA):NEXTAA
7540 PRINT"██EXITS:█"
7545 PRINT
7546 IFRF(39)=1THENPRINT"NORTH WEST"
7547 IFRF(40)=1THENPRINT"SOUTH WEST"
7550 INPUT ENTRY$
7555 IFENTRY$="NORTH WEST"ORENTRY$="NW"ANDRF(39)=1THEN7100:REM ROOM #39
7560 IFENTRY$="SOUTH WEST"ORENTRY$="SW"ANDRF(40)=1THEN7200:REM ROOM #40
7570 GOSUB100
7575 FORXX=1TODY:NEXTXX
7580 GOTO7500
10000 REM TITLE PAGE
10010 POKE53280,6:POKE53281,0
10015 POKE53265,PEEK(53265)AND191
10020 PRINT"██"
10030 PRINT"███";
10040 PRINTTAB(13)"MARK WILLIAMS"
10050 PRINT"█"TAB(16)"PRESENTS"
10060 PRINT"███";
10070 PRINT"
10080 PRINT"
10090 PRINT"
10100 PRINT"
10110 PRINT"
10120 PRINT"
10130 PRINT"
10140 PRINT"█"TAB(9)"██A ██SCI-FI█ ADVENTURE FOR"
10145 PRINTTAB(18)"█THE"
10150 PRINTTAB(17)"██C██M ██64"
10160 FORXX=1TO3500:NEXTXX
10170 PRINT"█"
10180 POKE53282,5:POKE53283,6:POKE53284,2
10190 POKE53265,PEEK(53265)OR64
10200 RETURN
```

Simon Rockman gives you all sorts of hints on how to become a sneaky programmer.

SNEAKY PROGRAMMING



ALL COMPUTERS AND computer systems have their quirks, the Commodore is no exception. A good programmer will know about and program around them. A sneaky programmer will exploit them to the full. This article is all about how to be a sneaky programmer.

Loading files

The Commodore tape system is slow and reliable, but it does have one feature few people know about. This is the ability to load a selected file by using part of the file name. To illustrate this imagine that you have a tape with three programs on it called "JOHN", "JAMES" and "GEORGE". To load the first program ("JOHN") you can just type in LOAD and press RETURN (or use SHIFT/RUN STOP). If you want to load the second file, bypassing the first you can type LOAD "JAMES" and the computer will search for that file finding, but not loading, "JOHN". What most people miss is that it is not necessary to type the whole name in; just LOAD "JA and pressing RETURN will perform the same function, similarly to load the last file, "GEORGE", with the tape wound to the beginning it is only necessary to type LOAD "G and press RETURN. This not only saves time but means that you can load a specific file even when you have forgotten the end of the name.

Auto-repeating

The 6502 inside a Commodore computer can put

any one of 256 values into any of the 65536 memory locations. Out of these sixteen million combinations one of the more useful is POKE650, 128 on the '64 and VIC. This enables auto-repeat on all keys.

Dating

In Britain we write the date in the order day/month/year, but in America they use month/day/year. The world ISO standard is however year/month/day and for computers this is the most logical method of storing a date if it has to be manipulated. This format allows the computer to sort the date. Take the 1st of January 1984 and 25th of September 1983. When used in the format 01011984 and 25081983 the date in 1983 has a greater value than the one in 1984. It would be possible to store the information like this and then chop the number up and sort all three bits separately but this would be slower and less efficient than having the dates held as 19840101 and 19830825 which follow in sequential order. When doing this it is important that you remember to pad out the spaces with zeros.

Toolkits

One of the commands often found in add-on toolkits is TRACE or TRON which displays the line being executed as the program runs. This is very useful when debugging code. The same sort of operation can be performed with a STOP command; before the program is run insert a STOP before and after the loops where you think the program is going wrong. Then when the program is run it will break in displaying the line number. You can continue using the CONT command providing that you do not cause any errors or alter the program. STOP has the added advantage that values can be inspected by using a PRINT while the computer is in direct mode. Once the program is debugged all the STOPs can be removed.

Rogue lines

When writing a program it is quite common for there to be a line which you are sure is right but seems to be causing the program to go wrong. The obvious solution is to delete the offending line to see if it was the culprit. If it wasn't the best bet is to re-enter the line and try elsewhere. This is a little counter-productive because you end up typing things in twice. A neater solution is to REM out the line. If you change the first three letters to REM then the line will do nothing while you test the program. When you want to use the line again you can just over type the REM with the original letters. Of course just taking bits out does not cure a programming problem but it can reduce confusion when several complicated operations are taking place at once.

Screen editing

The Commodore screen editor is one of the nicest ways of entering programs that anyone has come up with. What you enter on the screen is what you get. One thing it does lack is the ability to merge lines. Imagine a program with these lines

```
5100 IFA$="SNEAKY"THEN A$=""
5120 PRINT "SNEAKY HUH?"
```

This routine will always print "SNEAKY HUH?". To change it to only print the message when A$="SNEAKY" would mean re-typing the PRINT "SNEAKY HUH?" at the end of line 5100 and removing line 5120. The clever way to do this is to list the two lines and them move the cursor to the space between the 5120 and the PRINT command. Enter a colon (:) and move the cursor back one space onto the colon. Now hold down the shift and press insert twenty three times, until the colon goes just past the quotes from A$=" ". Still holding down the shift press RETURN. The cursor will move but the line will not have been entered. Now move the cursor up to the line above the 5100 which should still be on the screen. This line may have the LIST command above it. Type LIST again and the first part of the 5100 line will fall into place in front of the end of the 5120 line. Move the cursor up to this line and press RETURN over it. This operation sounds complicated when described on paper but if you try it, and get into the habit of using it, then you will save a lot time when changing your programs.

Protection

There are times when you want to protect a program from prying eyes. Most protection has to be done in machine code especially when you want to stop people from pirating your software. However, there are cases where just stopping a person from listing a line will suffice. On the Commodore 64 this can be done using a shifted L in a REM line. Just type

```
7 REM (Shifted L)
```

then when you list it the result will be:

```
/ REM
?SYNTAX ERROR
READY.
```

Please do not use this in any programs you send to "Your Commodore" because we will only have to remove these lines to use the program in the magazine.

IF7THEN100 may at first glance seem to be a syntax error, if Z what?. The meaning is really quite simple. The line has the same effect as saying IFZ 0THEN100 but saves three bytes. This is called a truth test and there are sneaky ways in which it is possible to use the equals sign. Try PRINT 1=1, this will give you the answer -1 which is the '64's way of saying true. PRINT=1 will return 0, meaning false. These operations should be used with care and carefully REMed because they can get confusing. They are a very useful and compact way of making a comparison and will work with strings as well as numbers.

Incorporating routines

When writing a large system it is common practice to have a set of standard subroutines which can be called from disc when needed. On a small system it is often desirable to do a similar thing but to incorporate the routine in each program. If you have a BASIC extension it is possible to SAVE the routine and then MERGE it into the main program. However, if you only have a standard machine it will be necessary to re-type the section of code each time it is needed. That is unless you are sneaky. With routines that are less than a screenful you can cheat. Load the small routine and list it to the screen. Make sure that you have five lines spare at the bottom and then load the main program. Without clearing the screen take the cursor to the first line of the routine to be incorporated and press RETURN. Do this for all the lines you want to merge. If you have more than twenty lines in the subroutine you can repeat this process in twenty line chunks.

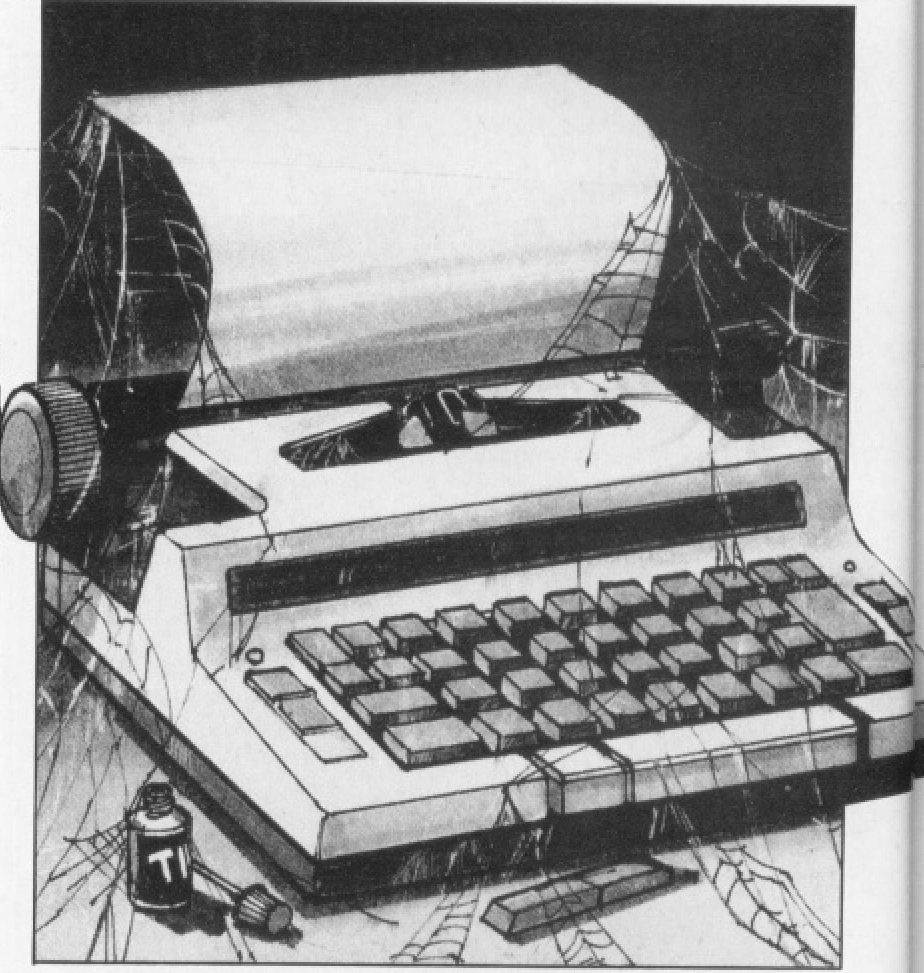
DATA STATEMENTS

Dataview

Colchester based software publishers, Dataview Wordcraft Limited, have just published an 8-page brochure claiming to unravel the mysteries of word processing, giving an overview of word processing, hardware and software, how it will save money or make money, with particular reference to their own Wordcraft software.

To obtain a free copy of this brochure, write to Amit Roy, Marketing Manager, Dataview Wordcraft Limited Radix House, East Street, Colchester, Essex CO1 2XB.

Dataview are also spreading the word on the continent having appointed distributors in Norway (Minor Mikrosystemer Norge A/S of Tonsberg), Belgium (Micro Belgium Application SPRL of Brussels) and the Netherlands (Intelligent Systems B.V. of Breda).



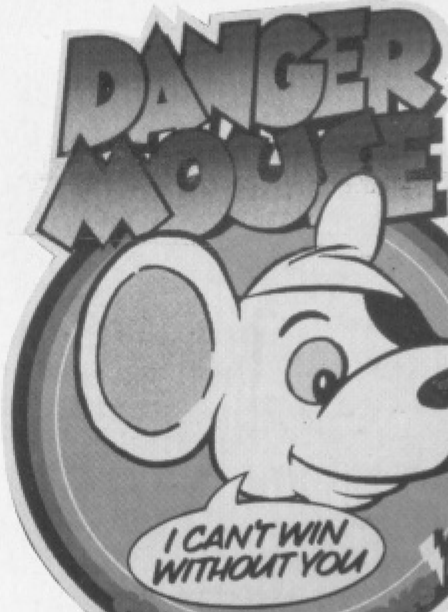
Screen graphics editor

Studio Software have recently released a new graphics designer package, Designer 64, which, supposedly, by relying on the standard Commodore graphics set, enables users to create impressive screen designs under program control which can then be incorporated easily into other programs with great effect.

As well as designers and planners, towards whom the program is largely directed, small businesses may also take advantage of the facilities provided by Designer 64.

The package is disc-based and the main program comes with seven demonstration design data files, a renumber utility program and a user guide.

Designer 64 is a Commodore Approved Product and is available on disc at £32.95. For further information, contact Studio Software at Rowan, Western Road, Jarvis Brook, Crowborough, East Sussex, TN6 3EY.



Danger Mouse

The popular TV cartoon character, Danger Mouse, is now starring in his own computer game. Creative Sparks, part of THORN EMI Computer Software, is releasing 'Danger Mouse in Double Trouble' on the Commodore 64.

Your aim is to help Danger Mouse, the world's greatest secret agent, to destroy ace villain, Baron Silas Greenback's, evil plan to dominate the world.

Creative Sparks have worked in close association with Cosgrove/Hall, producers of the Danger Mouse cartoon series, in developing the storyline and animation for 'Double Trouble'.

The game retails, on cassette, at £7.95 and should be available early October.

Creative Sparks are also giving the most skilful players of 'Danger Mouse in Double Trouble' the chance to enter a competition and win a trip to the Cosgrove/Hall studios to meet the creators of Danger Mouse. And you'll arrive there in style: by Rolls Royce and helicopter.

For further information, contact Gordon Reid, THORN EMI Computer Software, Thomson House, 296 Farnborough Road, Farnborough, Hants, Telephone: 0252-543333.

Craig Communications join A.V.S.

Dick Craig and David Giles have recently formed a new company to distribute and promote leisure software. Under the name of Craig Communications, they will initially market all the leisure products created by A.V.S., including Flight 015 and Whirlwind 15 on the VIC 20. They are also marketing System 15000 which you'll find reviewed in the Software Spotlight section of this month's Your Commodore.

Games to test your brainpower



On 21st August, Brighton-based Amplicon launched their first two 'Braingames', Electron Trail and Fame Quest. With this new range of games, Amplicon hope to provide the micro computer market with games which offer a challenge sufficient to entice the players back to the game again and again. Peter Wood of Amplicon reckons that his 'Braingames' are". . first and foremost easy to get into and great fun" but " . . .do need some brain work on the part of the players and so offer an element of compulsion".

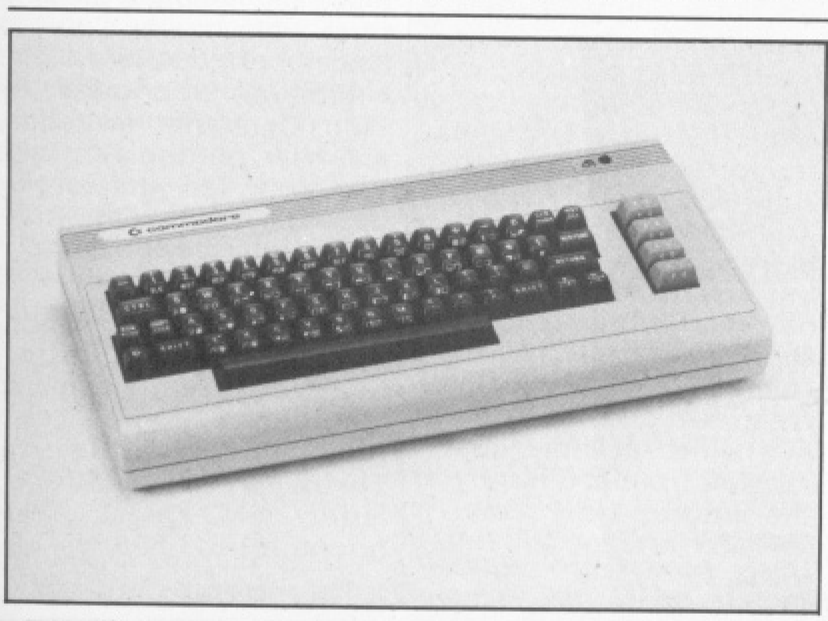
In Election Trail, the player finds himself campaigning on behalf of a party in the American election. As one player, you are campaigning for the Republicans while the computer backs the Democrats. Each state is worth a different number of points and has a handicap according to the likelihood of victory there. The aim is to win each state and then each region. Each player is initially presented with an opinion poll; he can then study his progress at various stages of the game by comparing new opinion polls against this original one. The campaigner can gain supporters through assorted means of publicity covering a range of prices such as media campaign, a rally, public debate, etc., depending on various factors such as history, hometown, etc. in each state. Voting works through from the top right hand state to the bottom with a recount demanded if final points are too close. The game concludes with the victorious party emerging to the sound of 'Stars and Stripes'.

Fame Quest claims to be a simply stratified game aiming to appeal to those who appreciate fantasies and roles. There are 10 grades each of which is attained by your knight exiting the castle, completing a quest and returning safely to the castle. The screen is divided into areas on a map with a castle in the top left and bottom right corners. Starting at the top, the knight encounters various challenges and the opportunity to pick up points of fame to reach the next grade. Although his options are limited at the lowest grade, the knight may buy weapons; depending on the key pressed (eg, 'e' for east) he will head in a different direction, encountering goodies and baddies — a dragon, wizard, old man and a damsel and choosing his method of approach — chat, flee or fight. With each successful encounter, the knight leaves the castle with a higher fame target.

Both these strategy games retail at £7.95 on cassette and £9.95 on disc and are available for the Commodore 64 from the end of August.

Two further Braingames will be introduced in September, Castle Fear and Flame Island.

For further information, contact Sheila Hart or Lisa Reuben at Public Image, 217-218 Tottenham Court Road, London W1P 9AF, Telephone: 01-580-6225.



New Passenger for Bubble Bus

Bubble Bus Software has taken over the marketing of business products from their offshoot — The Computer Room.

The first package to be marketed is Supernews, a newsagents delivery and accounting system, based on either the Commodore 64 or 8000 computers. This package, which has been selling consistently for two years, looks after up to 3000 deliveries, produces round lists, accounts, pre-order requirements and more. Its retail price is £499.00 excl. VAT.

Bubble Bus hope to expand its business dealer network to handle these products.

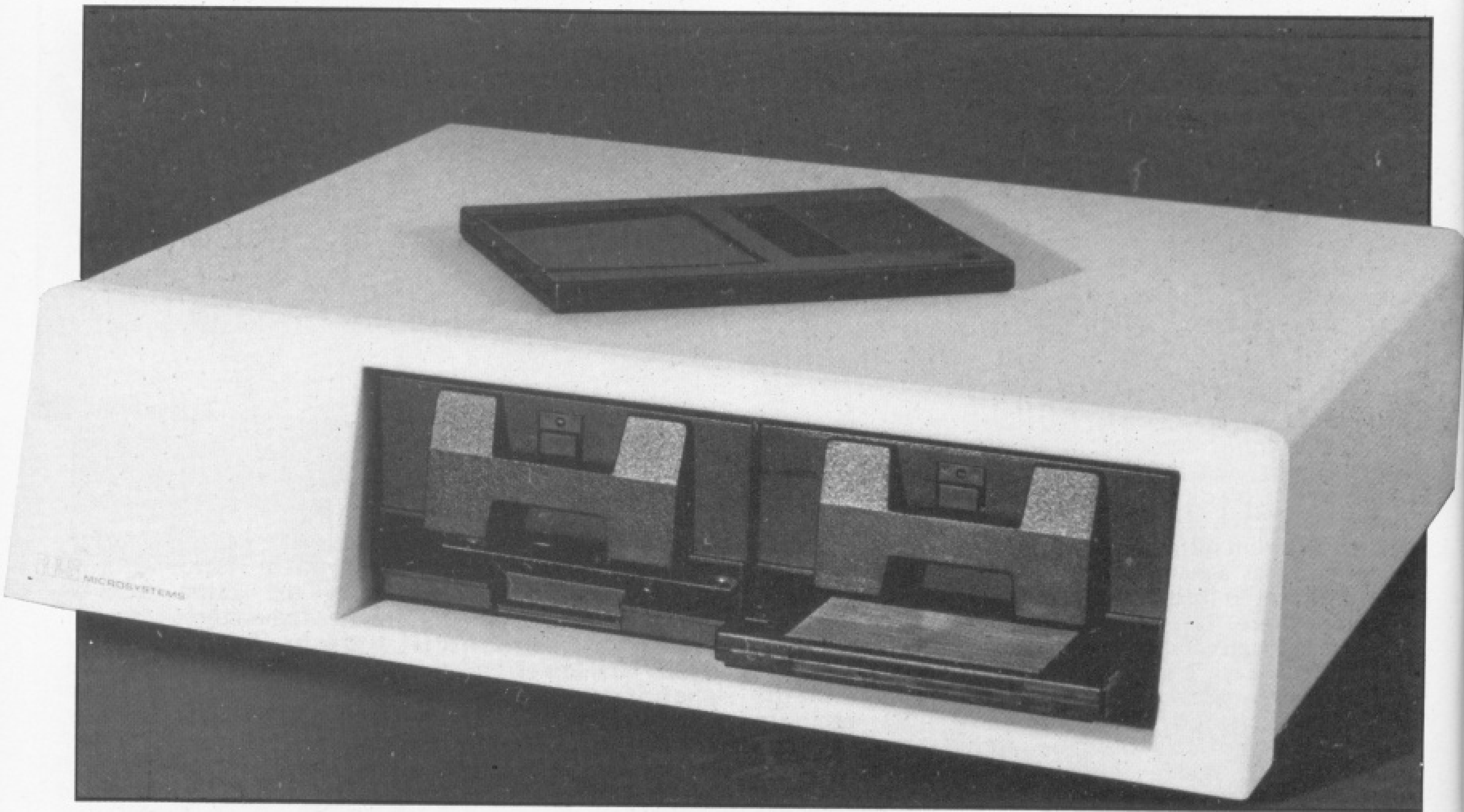
Cartridges from Apstor

News from Apstor Ltd is that, although slow off the mark, Beta 5 sales are set to equal, or even overtake, those of its big brother, Alpha 10. Reasons proffered are the relative compactness of the Beta 5 and the fact that its 10 Mbytes of storage are sufficient for most companies.

Apstor claim that both their systems combine the particular advantages of both cartridge-based hard discs and floppy disc systems, providing a solution to micro users who need big storage capacity with security back-up.

The units are tough and the Beta 5, like the Alpha 10, comes in a choice of a compact stacked or side-by-side configuration, depending on your space needs.

Apstor are based at Unit 5, Victoria Road Trading Estate, Portslade, Brighton, Sussex, BN4 1XQ. Telephone 0273 422512.



Channel 8 News

Channel 8 Software have recently signed reciprocal production agreements with Comm★data, an American company who will produce and market the Mysterious Adventure series for the Commodore 64. Accordingly, Channel 8 Software will now be able to offer some of America's best-selling Commodore 64 educational software. Each Educational Series tape contains four programs aimed at a specific age group and retails at £6.95 inc. VAT.

Also hot off the shelves of Channel 8 come two new games, Borzak and Time Zone.

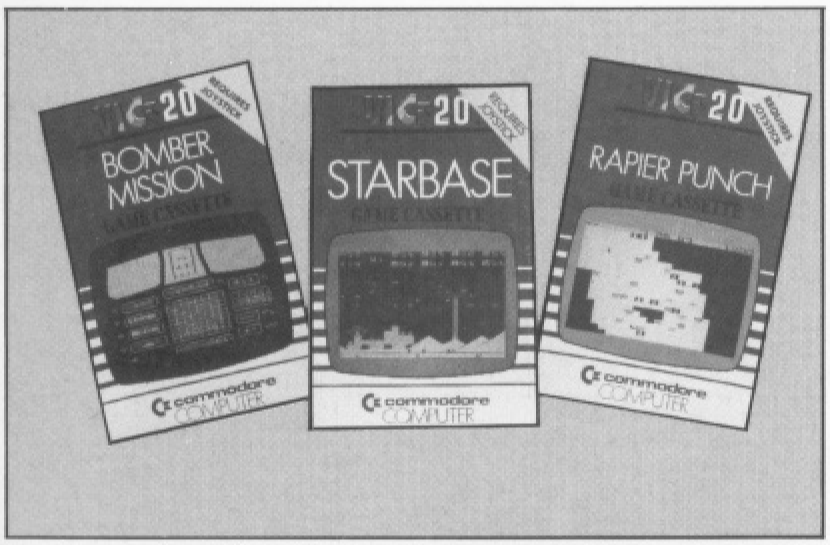
To give it its full title, Borzag. . .The Amazing Bug-Eyed Beastie From Betelgeuse, is a fast and furious arcade type game where Borzag, our anti-hero, is trying to get back to his space ship after, unintentionally, crash landing on earth en route back from a party on the planet Garulon. Borzak can run, jump, duck dodge and perform an unlimited number of anti-grav assisted jumps with either keyboard or joystick manipulating pits, pools, stone walls and various creatures on his journey home. Borzak is available on cassette and retails at £6.95 inc. VAT.

Time Zone, written in machine code, boasts a fine array of graphics, 20 levels of play, multi-sprite animation, 5 terrain types with perfect scrolling, on screen printouts, arcade quality sound, three speed star field and 'Ripple' High Score Table. It is a game for 1 or 2 players, using joystick or keyboard control, the object of which is to fight alien life forms that have changed to look like creatures or objects from five different time zones, ranging from pre-history to the future. Time Zone is available on tape and retails for £6.95.

Your Computer Christmas Fair

The Your Computer Christmas Fair will take place at Olympia 2, London, from November 30 to December 2, 1984. The exhibition, sponsored by Your Computer magazine, will have on display a large selection of microcomputers, peripherals, software and accessories.

For further information, contact the Exhibition Manager, Your Computer Christmas Fair, Reed Exhibitions, Surrey House, 1 Throwley Way, Sutton, Surrey SM1 4QQ. Telephone: 01-643-8040.



Games Galore from Commodore

New for the VIC 20 from Commodore come Bomber Mission, Rapier Punch and Starbase.

In Bomber Mission, as a World War II fighter bomber on a mission over hostile territory, your aim is to fly your aircraft to the target, bomb it and then return to base. But it's not as simple as it sounds. Your mission is beset with life's niggling little problems such as the time available to complete the mission, the amount of fuel needed, and the best type of weapon to use. And then there's the enemy: how good is enemy intelligence, can you detect enemy fighters on you radar screen before it's too late? Having fulfilled your mission by bombing your target and getting safely back to base, your ability as a pilot will be assessed on the amount of fuel and ammunition remaining and number of enemy aircraft destroyed. Bomber Mission's aircraft is controlled by a combination of joystick and function keys, and sound effects are incorporated into the program which runs on an expanded (16K) VIC 20 and retails at £4.99.

Commodore's other two new releases may be used on any unexpanded VIC 20.

In Rapier Punch, as a knight in a darkened room with only the areas you cross lit up, your aim is to find the hidden treasure chest before the timer reaches zero and to move on to the next level of the game. There are 100 levels in the game and details of your score, game level, number of lives remaining and time left to complete the game, are displayed on the top line of the screen. But your goal is hampered by spinning crosses, Dragons and Dragons' eggs set on ending each of your 3 lives and safeguarding the treasure. As you destroy these by firing daggers or running into them with your rapier, and finally achieve the ultimate in collecting the treasure, you accumulate points.

The aim of Starbase is to prevent a team of scientists, diligently preparing the surface of Planet XA2 for colonisation, from being captured by an alien force attacking the planet and to destroy all the alien spacecraft. This is achieved by patrolling each of the four quadrants of the planet (marked across the bottom of the screen) and by destroying the alien ships automatically once you have them in your sight. The top line of the screen tells you how many men you have left on the planet's surface and, once the attacking aliens have been destroyed and all your men are captured, the game ends.

Both Rapier Punch and Starbase need a joystick, include full sound effects and retail at £4.99.

Audiogenic in Gameland



Inspired by Lewis Carroll's novel, Audiogenic has launched Alice in Videoland. This storybook game contains several different scenarios, each relating to a specific section of the book and includes many of the old favourite characters such as the White Knights, Jabberwocky, Tweedledum and Tweedledee, the Caterpillar, the Red Queen and many others. The game follows the original story fairly closely starting with Alice's arrival at the entrance to the rabbit hole and continuing with her dilemmas with different sized doors and keys, bottles and cakes. The grinning Cheshire Cat and pipe smoking Caterpillar make an appearance in chapter two; the White Knights, Jaberwocky, Tweedledum and Tweedledee in chapter three and the Queen of Hearts and her croquet game in chapter four. The game ends when Alice has run out of croquet balls.

Controlled by a joystick and incorporaing fine graphics and music, Alice in Videoland sells for £12.95.

Audiogenic have also released Koala Pad which allows Commodore 64 users to produce full colour illustrations and drawings directly on screen with relative ease. The system includes a small and lightweight pad, cassette or disc-based software and an instruction manual. The user has a choice of colour, brush size and basic functions (such as line, circle, box, etc.). Drawings can be saved and recalled and other options such as copy, delete, change colour, merge images, are included.

Koala Pad is available on both disc and cassette and retails at £89.95.

For further information, contact: Audiogenic Ltd, P.O. Box 88, Reading, Berks; Telephone: 0734-664646.



Mikro 80 Cross-Assembler

Supersoft hope to have released their Z-80 cross-assembling version of the Commodore 64's excellent 6502 assembler, Mikro Assembler, by September 1st. Designed to run on the Commodore 64, Mikro 80 is being written in 6502 machine code but will assemble Z-80 opcodes rather than 6502 opcodes. If it is a success, Supersoft intend to follow Mikro 80 with versions for other processors.

Also being investigated by Supersoft is the idea of a direct cable link between the 64 and the Z-80 as the most likely means to transfer assembled code to target computers.

Interface from 3D

3D Digital Design and Development Ltd have released their latest microcomputer interface product, the GPIS. This scientific, industrial and educational interface is designed to work on the Commodore. It allows you to monitor up to eight analog signals with 12-bit resolution, and combines an integrating analog-to-digital convertor and a fast, successive approximation convertor enabling your Commodore to sample at rates up to 28 kilo-samples per second. Analog outputs are made available to give you proportional or three-term control, whilst the digital output enables your Commodore 64 to switch to up to eight loads with 50 volts at 400 mA each. The digital input facilitates the monitoring of eight binary signals or contact closures.

The GPIS is self-contained with integral power supply, connectors, and ribbon cable to your Commodore's expansion port. Full technical manual and a suite of demonstration programs are supplied with it. This interface retails at £700 but 3D offer a 25% discount to dealers and 50% off the price for demo units.

Soaring Commodore sales

According to the 1984 BIS-Pedder Annual Census of Information Processing, Commodore have sold so many machines in 1983 that by value they are third in the table of market leaders, behind IBM and ICL but ahead of DEC and Sinclair.

Although not in the same league as IBM who captured a huge 23.7% of the market, Commodore's market share jumped from 3.6% in 1982 to 6.3% in 1983, slightly behind ICL's 7.2% DEC gained only 5.1% and Sinclair 4.1%.

MARKET LEADERS IN VALUE OF COMPUTER SHIPPED IN 1983

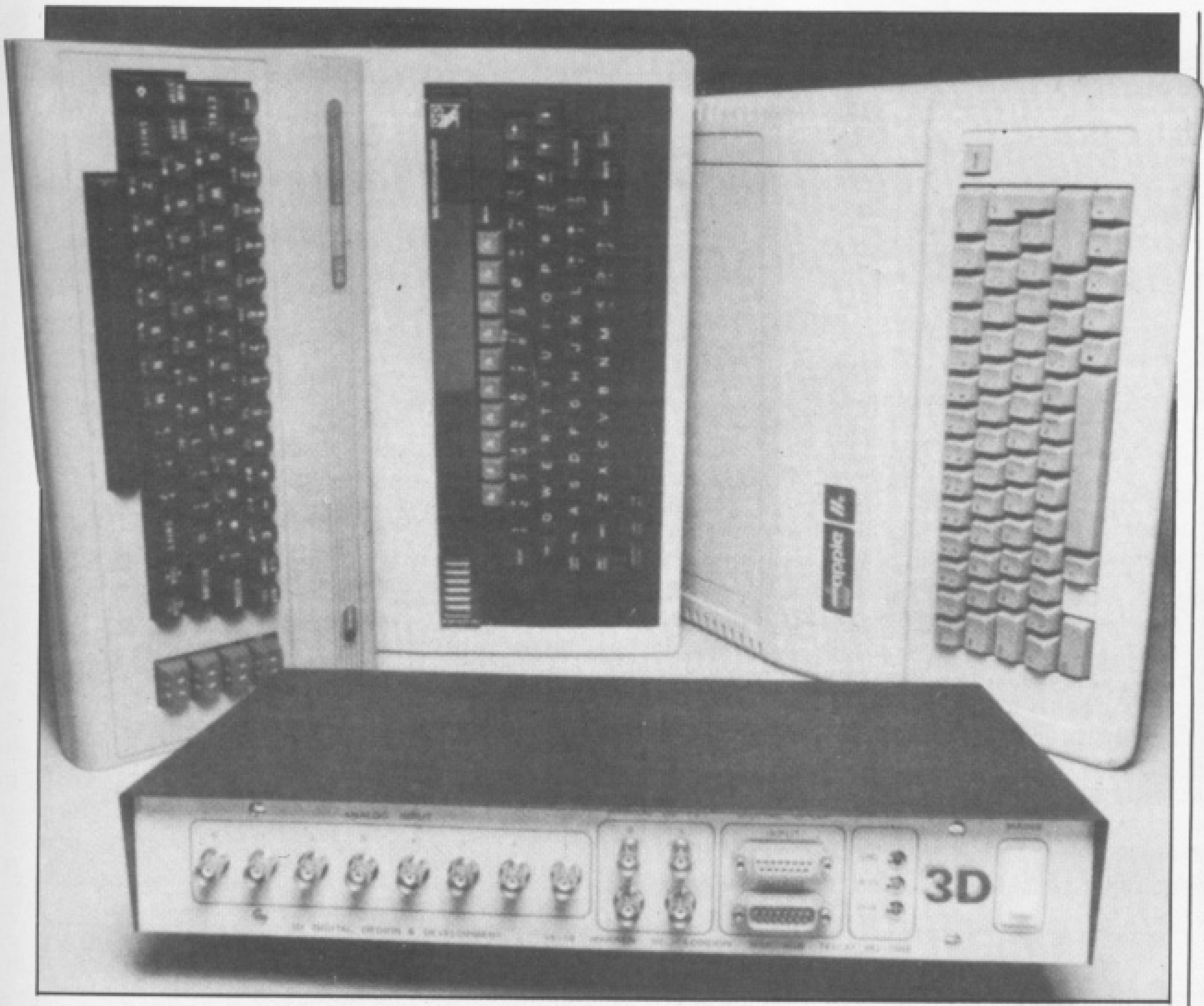
Company	percentage 1983	value shipped in 1982
IBM	23.7	28.3
ICL	7.2	11.0
Commodore	6.3	3.6
Digital	5.1	5.3
Sinclair	4.1	1.3
one percent =	£22.4m	£16.9m

Leap forward for Cheetah

Cheetah Marketing has not only taken over the sole manufacturing and marketing rights to Interpod, the Commodore 64 and VIC 20 interface, but has also reduced its price to £59.95.

Interpod provides Commodore users with full RS232 and IEEE interface facilities enabling users to access all Commodore business peripherals and take advantage of assorted independent products such as hard discs, printers, etc.

Parc Electronics, who originally manufactured the Interpod on behalf of Oxford Computer Systems Ltd., recently acquired Cheetah Marketing. Oxford Computers' recent problems have given Cheetah Marketing the opportunity to take Interpod under their wing.



Argus hits the small screen

Argus Press Software have just announced their first plans for the Autumn. Their latest release in their Mind Games series is 'American Football', a full graphics simulation for one or two players available for the CBM 64. With the game comes a book explaining 'all you ever wanted to know but were too afraid to ask' about American Football.

Argus Press are promoting their products through an extensive advertising campaign in the press and on TV. They already have a TV advert for their American Football program, booked for the Superbowl final in January. Argus Press Tape Magazines will also be supported by TV adverts from mid-September to November. The zany characters from the 'Young Ones' will do the voice over for the adverts. The Clever Cloggs series of programs will also be extensively advertised, in major consumer magazines.

Argus Software are at No. 1 Golden Square, London W1R 3AB.

Camden Computers

Commodore Business Machines (UK) Ltd. have announced their Commodore Dealership of the Year: Birmingham-based Camden Computers. In the year ended June 1984, Camden Computers achieved over £800,000 of sales of Commodore business systems.

Camden Computers, formed in 1971, has been a Commodore Approved Dealer since Commodore's appearance in the UK during the mid '70s. With a nationwide, thousand plus user-base, Camden has become one of Commodore's most successful retailers. Brothers Derek and Ronald Bailey, sole directors of Camden Computers, were recently presented with a cup by Commodore to mark their achievement. Naturally, they are delighted with the award.

"We have been with Commodore since day one", said Ron, "so it's particularly gratifying to reap the rewards of our commitment. We supply many of the largest and most prestigious companies in the West Midlands with Commodore machines, from the earliest PETs to the latest 8000 series computers, and for us it's been an extremely fruitful relationship".

International Programming Competition

On Saturday 20th October, the first ever European Heat in the thirteen year history of the ACM's (Association of Computing Machinery) International Programming Competition will take place at Thames Polytechnic, London. The contest is being sponsored by Commodore Business Machines (UK) Ltd. and Thames Polytechnic, and is being organised with the full support of the British Computer Society.

The competition takes the form of a team of up to four undergraduate and postgraduate students solving a set of six programming problems in as short a time as possible. Solutions may be programmed in either Pascal or Fortran, using Commodore 8000 computers. The competition will last for six hours. The contest will be followed by a reception and the announcement of the two winning teams who will go on to represent the European region at the Final in New Orleans next March. Included amongst the panel of judges will be Professor Wolff of BBC Television's 'Great Egg Race'. Spectators are admitted free of charge.

The Company's UK General Manager, Howard Stanworth, believes that Commodore should be involved with the competition due to their position in the forefront of British education. He states that his company ".. intend to continue investing heavily in education in this country and this is just one of the forms that investment will take". Through a victory in the International Final of the competition, he hopes to prove that ". . .Europe still leads the world in computer skills and programming creativity".



SAMMY THE SLUG

Score points as you help Sammy the Slug meander through a selection of mazes, picking up jars of cabbage on his way, with this game from F G Tout.

YOUR TASK IS TO GUIDE Sammy through 14 waves collecting jars of cabbage. Each screen consists of a maze of walls. You can walk on the walls but not through them. Control is with the joystick: press 'fire' to jump.

To proceed from the start, hit 'F1': for a random maze hit 'R'. The selection of mazes includes such gems as 'Beam me Up' and 'Fairground'. You have 3 lives on each maze and have to collect all the cabbages to move onto the next maze. If you lose a life, all the cabbages re-appear.

Points are collected for retrieving the jars of cabbage.

Sammy the slug uses approximately 30K of memory when run, as the CBM-64 can only see 16K at one time, it has been necessary to move the screen and VIC 2 chip, this is O.K. unless run/stop and restore are used which causes the program to crash.

Type in Part 1 then Save.

Type in Part 2 then Save. Remember to Save Part 1 and Part 2 separately until certain that it is working.

You can stop the program and list it, but do NOT use RESTORE key.

Save before running and to get the screen back after RUN/STOP-RESTORE type: POKE648,4.

Program Listing

```
0 POKE53265,PEEK(53265)AND239
1 POKE51,255:POKE52,127:POKE55,255:POKE56,127:POKE56578,PEEK(56578)OR3
2 POKE56576,((PEEK(56576)AND252)OR1)
3 POKE53272,((PEEK(53272)AND15)OR48):POKE53272,(PEEK(53272)AND240)OR0
4 POKE648,140
20 POKE56334,PEEK(56334)AND254
30 POKE1,PEEK(1)AND251
40 FORG=0TO64*8:POKE32768+G,PEEK(53248+G):NEXT
50 POKE1,PEEK(1)OR4
60 POKE56334,PEEK(56334)OR1
100 READA%:T=T+1:IFA%<>-1THENPOKE32768+G+T,A%:GOTO100
104 DATA0,0,0,0,0,0
105 DATA247,247,0,223,223,0,251,0
111 DATA130,68,104,144,251,127,15,3,0,0,48,252,255,255,255,231
112 DATA0,0,0,0,192,240,252,231,56,68,146,186,146,68,56,0
120 DATA0,0,0,0,3,15,63,231,0,0,24,63,255,255,255,231
121 DATA65,34,22,9,223,254,240,192,170,195,36,90,153,36,195,0
124 REM*****BEETLE(J/K)****
125 DATA3,29,51,111,255,44,82,145,136,224,244,255,255,108,146,9
130 DATA85,195,36,153,90,36,195,24,224,56,254,127,192,0,192,0
135 DATA0,192,0,192,32,248,126,255
136 REM*****UP BELT ↑*****
140 DATA56,84,16,16,126,68,102,126
145 DATA60,126,90,60,28,28,0,28,0,24,60,110,118,122,126,126
150 DATA130,84,56,84,169,170,170,170
155 DATA218,218,146,254,254,218,146,254
160 DATA255,129,253,255,129,153,129,255,170,170,239,255,255,239,170,170
165 DATA255,255,153,153,255,255,153,153
170 DATA40,146,84,58,127,255,127,58,0,4,40,18,164,233,255,233
175 DATA0,141,82,47,114,241,118,58
180 DATA255,255,129,129,129,129,255,255,24,62,102,223,215,94,248,96
185 DATA2,7,14,60,56,216,108,32,0,126,125,118,112,104,80,32
190 DATA24,24,60,60,60,66,16,16
195 DATA255,195,102,60,60,102,195,255
248 DATA-1
252 FORN=0TO79:READA:POKE32768+384+N,A:NEXT
253 FORN=0TO207:READA:POKE32768+8+N,A:NEXT
350 DATA126,70,74,82,102,70,126,0,8,8,8,24,24,24,24,0
352 DATA126,2,2,126,96,96,126,0
354 DATA126,2,2,62,6,6,126,0,64,68,68,126,12,12,12,0
356 DATA126,64,64,126,6,6,126,0,64,64,64,126,98,98,126,0
358 DATA126,66,66,6,6,6,6,0,126,66,66,126,98,98,126,0
360 DATA126,70,70,126,2,2,2,0
361 REM*****LETTERS*****
362 DATA126,66,66,126,98,98,98,0,124,68,68,126,98,98,126,0
364 DATA126,66,66,96,98,98,126,0,126,66,66,98,98,98,126,0
366 DATA126,64,64,124,96,96,126,0
368 DATA126,64,64,124,96,96,96,0,126,66,64,102,98,98,126,0
370 DATA66,66,66,126,98,98,98,0,16,16,16,24,24,24,24,0
371 REM*****J*****
372 DATA2,2,2,6,70,70,126,0,66,66,66,124,98,98,98,0
374 DATA64,64,64,96,96,96,126,0,126,74,74,106,106,106,106,0
375 REM*****N*****
376 DATA98,82,74,70,98,98,98,0,126,66,66,70,70,70,126,0
378 DATA126,66,66,126,96,96,96,0,126,66,66,106,106,126,8,0
380 DATA124,68,68,126,98,98,98,0,126,98,96,126,2,66,126,0
382 DATA126,16,16,24,24,24,24,0,66,66,66,98,98,98,126,0
384 DATA66,66,66,102,36,24,24,0,74,74,74,106,106,126,126,0
386 DATA66,66,126,24,126,66,66,0,66,66,66,126,24,24,24,0
388 DATA126,66,68,8,18,34,126,0
600 POKE53265,PEEK(53265)OR16:PRINT"J"
610 PRINT"XXXXXXXXXXXPLEASE PRESS SHIFT/RUN.STOP TO LOAD PT 2"

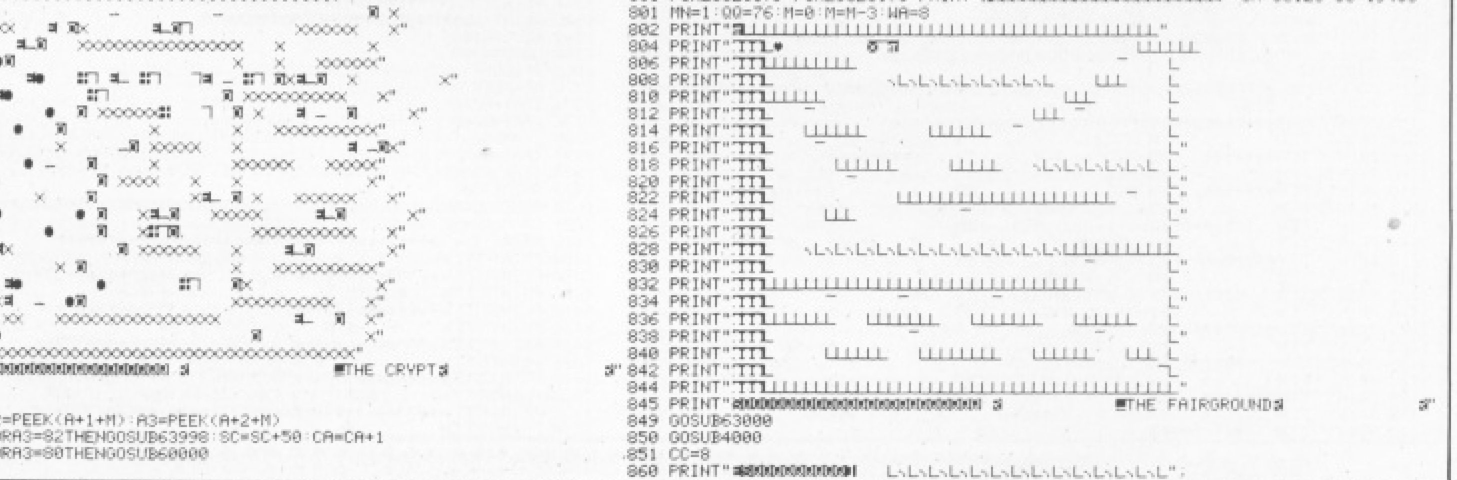
READY.
```

```
5 POKE650,255:GOSUB55000:QQ=65:Z2=150
10 LI=3:QQ=66:CX=0:WA=0
100 POKE53280,2:POKE53281,0:PRINT"JXXXXXXXXXXXXXXXXXXXXXXXXX":M=221:R=35:CA=0
101 WA=WA+1:QQ=65
102 PRINT"X****************************************"
104 PRINT"TT*                                    ## **"
106 PRINT"TT**********    ****************************"
108 PRINT"TT*     _           *           *          *"
110 PRINT"TT*****************   ***********   ****    *"
112 PRINT"TT*                                  _      *"
114 PRINT"TT*************************************     *"
116 PRINT"TT*                                         *"
118 PRINT"TT*   *** _     _     _ *** _     _ ******"
120 PRINT"TT*   **************************************"
122 PRINT"TT*                                         *"
124 PRINT"TT**********************************    ****"
126 PRINT"TT*-                                        *"
128 PRINT"TT*****   ********************************"
130 PRINT"TT*    *          *       *                 *"
132 PRINT"TT*    ********* _ *   *   *   **********   *"
134 PRINT"TT*    *          *   * _ *            *    *"
136 PRINT"TT*    * _ ********   *   **********   *    *"
138 PRINT"TT* _ *               *                * _ *"
140 PRINT"TT*    **********************************   *"
142 PRINT"TT*                                         *"
144 PRINT"TT*****************************************"
145 PRINT"XXXXXXXXXXXXXXXXXXXXXXXXX S                XTHE  DENS"
149 GOSUB63000
150 GOSUB4000:QQ=65
155 PRINTTAB(R)"XXXXXXXXXXXXXXXXXX~  "
170 R=R-1.5:IFR<3THENPRINTTAB(R)"]M  ":R=35
172 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
174 IFA1=82ORA2=82ORA3=82THENGOSUB63998:SC=SC+50:GA=CA+1
176 IFA1=74ORA2=74ORA3=74THENGOSUB60000:SC=SC+100
178 IFA1=75ORA2=75ORA3=75THENGOSUB60000:SC=SC+100
195 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO200
197 IFA3=83ANDCA=13ANDPL=1THENGOSUB50000:GOSUB5000: GOTO16000
198 IFA3=83ANDCA>12THENWA=1:GOSUB50000:GOSUB5000:GOTO200
199 GOTO150
200 POKE53280,5:POKE53281,11:PRINT"JXXXXXXXXXXXXXXXXXXXXXXXXXX":M=165:CA=0:WA=2
201 R=35:R1=3:QQ=84
202 PRINT"X||||||||||||||||||||||||||||||||||||||||"
204 PRINT"TTT   X       | _     | _     | _      XX |"
206 PRINT"TTT    |||||    |||||    |||||    |||||    ||||"
208 PRINT"TTT    |                                     |"
210 PRINT"TTT    ||   |||||    |||||    |||||    ||||||"
212 PRINT"TTT                                          |"
214 PRINT"TTT|    |||||    |||||    |||||    |||||    |||"
216 PRINT"TTT                                          |"
218 PRINT"TTT|||||||||||||||||||||||||||||||||||||    |"
220 PRINT"TTT           _ _ _ _ _ _ _ _                |"
222 PRINT"TTT    ||||||||||||||||||||||||||||||||||||||"
224 PRINT"TTT                                          |"
226 PRINT"TTT|||||||||||||||||||||||||||||||||||||    |"
228 PRINT"TTT                                  |       |"
230 PRINT"TTT    |||||||||||||||||||||||||||   |    ||"
232 PRINT"TTT                                          |"
234 PRINT"TTT  _ |   |       _|       _|    |   ||    |"
236 PRINT"TTT    |  _ |||||||||||||||||||||    |       |"
238 PRINT"TTT    |                             |    ||"
240 PRINT"TTT    ||||||||||||||||||||||||||||||       |"
242 PRINT"TTT                                          |"
244 PRINT"TTT||||||||||||||||||||||||||||||||||||||||"
245 PRINT"XXXXXXXXXXXXXXXXXXXXXXXXX S                XTHE MAZES"
249 GOSUB63000
250 GOSUB4000
255 PRINTTAB(R)"XXXXXXXXXXXX? "
256 PRINTTAB(R1)"XXXX  ?"
257 R=R-1:IFR<3THENPRINTTAB(R)"TTTTT  ":R=35
258 R1=R1+1.5:IFR1>34THENPRINTTAB(R1)"]  ":R1=3
272 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
274 IFA1=82ORA2=82ORA3=82THENGOSUB63998:SC=SC+50:CA=CA+1
```

Program Listing

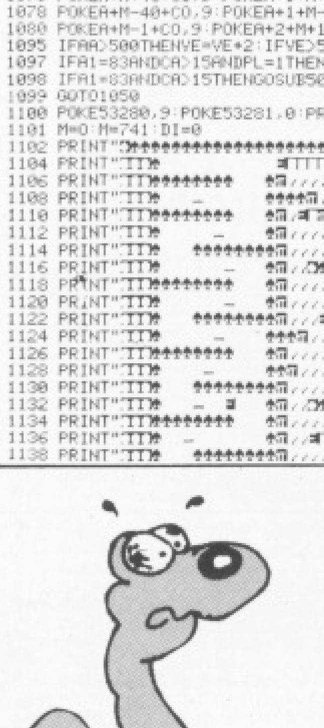
```
276 IFA1=63ORA2=63ORA3=63THENGOSUB60000:SC=SC+100
295 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO300
297 IFA3=83ANDCA>12ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
298 IFA3=83ANDCA>12THENGOSUB50000:GOSUB5000:GOTO300
299 GOTO250
300 POKE53280,4:POKE53281,3:PRINT"[graphics]":CC=9:M=16:WA=3
301 CA=0:QQ=65
302 PRINT"♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
304 PRINT"TT♣     _     ♣     _     ♣ _     ♣   _ ♣♣"
306 PRINT"TT♣   ♣♣♣♣♣   ♣   ♣♣♣♣♣   ♣   ♣   ♣   ♣♣♣♣"
308 PRINT"TT♣     _ ♣   ♣ _     ♣   ♣_  ♣   ♣   _ ♣"
310 PRINT"TT♣♣♣♣♣   ♣   ♣♣♣♣♣   ♣   ♣   ♣   ♣   ♣♣♣♣"
312 PRINT"TT♣   _   ♣   ♣       ♣       ♣       _♣"
314 PRINT"TT♣   ♣♣♣♣♣   ♣   ♣♣♣♣♣   ♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
316 PRINT"TT♣       ♣       ♣_      ♣           _♣"
318 PRINT"TT♣♣♣♣♣   ♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣   ♣   ♣♣♣♣♣♣♣♣"
320 PRINT"TT♣  _    ♣           ♣       ♣       _ ♣"
322 PRINT"TT♣   ♣♣♣♣♣   ♣♣♣♣♣   ♣   ♣♣♣♣♣♣♣♣♣   ♣♣♣♣"
324 PRINT"TT♣       ♣       ♣   ♣       ♣       _ ♣"
326 PRINT"TT♣♣♣♣♣   ♣♣♣♣♣   ♣   ♣♣♣♣♣   ♣   ♣♣♣♣♣♣♣♣"
328 PRINT"TT♣ _       _♣    ♣           ♣       _ ♣"
330 PRINT"TT♣   ♣♣♣♣♣♣♣♣♣   ♣♣♣♣♣   ♣♣♣♣♣♣♣♣♣   ♣♣♣♣"
332 PRINT"TT♣       ♣       ♣_      ♣_          _ ♣"
334 PRINT"TT♣♣♣♣♣   ♣   ♣♣♣♣♣♣♣♣♣♣♣♣♣   ♣♣♣♣♣♣♣♣♣♣♣♣"
336 PRINT"TT♣_      ♣       ♣      _♣   ♣       _♣"
338 PRINT"TT♣   ♣♣♣♣♣♣♣♣   ♣♣   ♣♣♣♣♣   ♣   ♣   ♣♣♣♣"
340 PRINT"TT♣       ♣       ♣       ♣       ♣       ♣"
342 PRINT"TT♣   ♣ _     ♣       ♣   ♣   ♣   ♣   ♣ ♣"
344 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
345 PRINT"[graphics]            DISAPPEARING TRICK        "
349 GOSUB63000
350 GOSUB4000
372 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
374 IFA1=82ORA2=82ORA3=82THENGOSUB63998:SC=SC+50:CA=CA+1:CC=CC+1
395 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO400
397 IFA3=83ANDCA>24ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
398 IFA3=83ANDCA>24THENGOSUB50000:GOSUB5000:GOTO400
399 GOTO350
400 POKE53280,6:POKE53281,0:PRINT"[graphics]":CC=8:WA=4:M=0:CA=0
401 QQ=65
402 PRINT"♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
404 PRINT"TT♣♣■                                    ♣
406 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣                        ♣
408 PRINT"TT♣                  ♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣
410 PRINT"TT♣   _♣♣♣♣♣♣♣♣♣♣♣♣♣     _  _ _          ♣
412 PRINT"TT♣                    ♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣
414 PRINT"TT♣    _    ♣♣♣♣♣♣♣♣♣♣♣       ♣♣♣♣♣♣♣♣♣♣♣♣♣
416 PRINT"TT♣       _   ♣♣♣♣♣♣♣♣♣♣♣                ♣
418 PRINT"TT♣♣♣♣♣       _           ♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣
420 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣       ♣♣♣♣♣♣♣♣♣♣♣♣         ♣
422 PRINT"TT♣                 ♣♣♣♣♣♣♣♣♣♣♣♣         ♣
424 PRINT"TT♣ ♣♣♣♣♣♣♣♣♣♣♣          _       _♣♣♣♣♣♣♣♣
426 PRINT"TT♣      _       _♣♣♣♣♣♣♣♣♣♣♣♣           ♣
428 PRINT"TT♣      ♣♣♣♣♣♣♣♣♣♣♣                  ♣♣♣♣
430 PRINT"TT♣                 ♣♣♣♣♣♣♣♣♣♣♣♣         ♣
432 PRINT"TT♣        ♣♣♣♣♣♣♣♣♣♣♣♣                  ♣
434 PRINT"TT♣♣♣♣♣                  ♣♣♣♣♣♣♣♣♣♣♣     ♣"
436 PRINT"TT♣          _  ♣♣♣♣♣♣♣♣♣♣♣              ♣"
438 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣                 ♣♣♣♣♣♣♣♣♣♣♣♣
440 PRINT"TT♣                ♣♣♣♣♣♣♣♣♣♣♣♣          ♣
442 PRINT"TT♣                 _       _            ♣"
444 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
445 PRINT"[graphics]            THE DARK ROOM        "
449 GOSUB63000
450 GOSUB4000
472 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
474 IFA1=82ORA2=82ORA3=82THENGOSUB63998:SC=SC+50:CA=CA+1
476 POKEA+M+40+CO,9:POKEA+1+M+40+CO,9:POKEA+2+M+40+CO,9
478 POKEA+M-40+CO,9:POKEA+1+M-40+CO,9:POKEA+2+M-40+CO,9
480 POKEA+M-1+CO,9:POKEA+2+M+1+CO,9
495 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO500
497 IFA1ANDCA>14ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
498 IFA1ANDCA>14THENGOSUB50000:GOSUB5000:GOTO500
499 GOTO450
500 POKE53280,11:POKE53281,0:PRINT"[graphics]":M=47:WA=5:CA=0
501 QQ=86
502 PRINT"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
504 PRINT"TTX[graphics]    _      _           X"
506 PRINT"TTXXXXXXX   [graphics]   [graphics]   XXXXXX   X"
508 PRINT"TTX      [graphics]  XXXXXXXXXXXXXXXXX   X     X"
510 PRINT"TTX   [graphics]           X   X   XXXXXX"
512 PRINT"TTX        [graphics]   X     X"
514 PRINT"TTX     [graphics]   [graphics]   XXXXXXXXXX   X"
516 PRINT"TTX[graphics]   ●   XXXXXX   X   _   X"
518 PRINT"TTX [graphics]  ●         X   X   XXXXXXXXXX"
520 PRINT"TTX[graphics] ●       X   XXXXX   X   [graphics]X"
522 PRINT"TTX [graphics]  ● _   X      X   XXXXXX   XXXXXX"
524 PRINT"TTX [graphics] ●    XXXX   X   X   X"
526 PRINT"TTX  [graphics]   X   X   X   XXXXXX   X"
528 PRINT"TTX [graphics]  ●   ●   X   XXXXX   [graphics]   X"
530 PRINT"TTX [graphics]    ●   [graphics]   XXXXXXXXXX   X"
532 PRINT"TTX   [graphics]X      XXXXXX   X   [graphics]   X"
534 PRINT"TTX [graphics]     X   X   XXXXXXXXXXX"
536 PRINT"TTX    [graphics]   ●   [graphics]   X"
538 PRINT"TTXXXXXX[graphics] _ ●   XXXXXXXXXX   X"
540 PRINT"TTX     XX   XXXXXXXXXXXXXXXXX   [graphics]   X"
542 PRINT"TTX   [graphics]          [graphics]          X"
544 PRINT"TTXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
545 PRINT"[graphics]            THE CRYPT        "
549 GOSUB63000
550 GOSUB4000
572 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
574 IFA1=82ORA2=82ORA3=82THENGOSUB63998:SC=SC+50:CA=CA+1
578 IFA1=80ORA2=80ORA3=80THENGOSUB60000
580 IFA1=81ORA2=81ORA3=81THENGOSUB60000
595 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO600
597 IFA1=83ANDCA>15ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
598 IFA1=83ANDCA>15THENGOSUB50000:GOSUB5000:GOTO600
599 GOTO550
600 POKE53280,2:POKE53281,10:PRINT"[graphics]":WA=6:CA=0:CC=0
601 M=741:QQ=65
602 PRINT"♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
604 PRINT"TT♣        ■_                _   _     _♣
606 PRINT"TT♣♣♣♣♣   ♣♣   ♣ ■_ ♣   ♣♣♣   ♣♣♣♣♣♣♣♣   ♣♣
608 PRINT"TT♣    ■_  _♣   ♣ . ♣                    ♣
610 PRINT"TT♣  ♣♣♣♣♣♣♣♣♣♣♣♣   ♣ ■_ ♣♣♣♣♣♣♣♣♣ ■ _ _    ♣
612 PRINT"TT♣           ■_ ♣ ♣   ♣   ♣■ _  _ ♣         ♣
614 PRINT"TT♣  ♣♣♣♣♣♣♣   ♣ ■_ ♣♣♣♣♣       ♣♣♣♣♣♣♣♣♣   ♣
616 PRINT"TT♣                  ■_          _        ♣
618 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣
620 PRINT"TT♣   ■_      ♣♣♣♣♣♣♣♣♣♣     ■_     ♣♣♣♣♣   ♣
622 PRINT"TT♣   ♣♣♣♣ ■_                ♣♣♣♣♣♣♣♣♣      ■_ ♣
624 PRINT"TT♣            ♣♣♣ ■_ ♣♣♣              ♣♣   ♣
626 PRINT"TT♣♣♣♣♣♣♣   ♣              ■_   ♣♣♣♣♣   ■_    ♣
628 PRINT"TT♣            ■_♣♣♣♣♣♣♣   ♣♣♣            ♣♣♣♣♣♣
630 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣
632 PRINT"TT♣■_             _    _               _♣
634 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣   ♣♣♣♣♣♣♣♣♣♣♣   ♣♣♣♣♣♣♣♣♣♣♣
636 PRINT"TT♣              ■_             _      ♣
638 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣
640 PRINT"TT♣   ♣■_      ♣        ♣■_      ♣ ■_      ♣♣♣♣
642 PRINT"TT♣         ♣    ■_ ♣        ♣ ■_     ♣     [graphics]"
644 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
645 PRINT"[graphics]            BEAM ME UP        "
649 GOSUB63000
650 GOSUB4000
651 CC=11
655 IFCA=10THENM=421:CA=11
656 IFCA=16THENM=214:CA=17
657 IFCA=20THENM=26:CA=21
672 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
674 IFA1=82ORA2=82ORA3=82THENGOSUB63998:SC=SC+50:CA=CA+1
695 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO700
697 IFA3=83ANDCA>24ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
698 IFA3=83ANDCA>24THENGOSUB50000:GOSUB5000:GOTO700
699 GOTO650
700 POKE53280,11:POKE53281,6:PRINT"[graphics]"
701 M=0:WA=7:M=0:CA=0:QQ=90
702 PRINT"♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦"
704 PRINT"TT♦                                      ♦"
706 PRINT"TT♦♦♦♦ ■_] ♦♦♦   ♦♦♦ ■_] ♦♦♦   ♦♦♦ ■_] ♦♦♦   ♦♦♦"
708 PRINT"TT♦                                      ♦"
710 PRINT"TT♦♦♦   ♦♦   ♦♦   ♦♦   ♦♦   ♦♦   ♦♦   ♦♦♦♦"
712 PRINT"TT♦                      ■_]             ♦"
714 PRINT"TT♦♦   ♦   ♦ ■_] ♦   ♦   ♦   ♦   ♦   ♦   ♦♦♦"
716 PRINT"TT♦                                      ♦"
718 PRINT"TT♦♦♦   ♦♦   ♦♦   ♦♦   ♦♦   ♦♦   ♦♦   ♦♦♦♦"
720 PRINT"TT♦                                  ■_]♦"
722 PRINT"TT♦♦♦♦   ♦♦♦   ♦♦♦   ♦♦♦   ♦♦♦   ♦♦♦   ♦♦♦"
724 PRINT"TT♦■_]                                   ♦"
726 PRINT"TT♦♦♦♦♦   ♦♦♦♦   ♦♦♦♦   ■_]♦♦♦♦   ♦♦♦♦   ♦♦♦♦"
728 PRINT"TT♦                                      ♦"
730 PRINT"TT♦♦♦♦♦♦   ♦♦♦♦♦   ♦♦♦♦♦   ♦♦♦♦♦   ♦♦♦♦♦♦♦"
732 PRINT"TT♦          ■_]                         ♦"
734 PRINT"TT♦♦♦♦♦   ♦♦♦♦   ♦♦♦♦   ♦♦♦♦   ♦♦♦♦   ♦♦♦♦"
736 PRINT"TT♦                      ■_]             ♦"
738 PRINT"TT♦♦♦♦   ♦♦♦   ♦♦♦   ♦♦♦   ♦♦♦   ♦♦♦■_]   ♦♦♦"
740 PRINT"TT♦■_]                                [graphics]"
742 PRINT"TT♦♦♦   ♦♦   ♦♦   ♦♦   ♦♦   ♦♦   ♦♦   ♦♦♦♦"
744 PRINT"TT♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦"
745 PRINT"[graphics]            STEPPING STONES        "
749 GOSUB63000
750 GOSUB4000
751 CC=8
772 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
774 IFA1=82ORA2=82ORA3=82THENGOSUB63998:SC=SC+50:CA=CA+1
795 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO800
797 IFA3=83ANDCA>11ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
798 IFA3=83ANDCA>11THENGOSUB50000:GOSUB5000:GOTO800
799 GOTO750
800 POKE53280,9:POKE53281,0:PRINT"[graphics]":CN=36120:CO=19456
801 MN=1:QQ=76:M=0:M=M-3:WA=8
802 PRINT"LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL"
804 PRINT"TTL●          [graphics]                       LLLLLL
806 PRINT"TTLLLLLLLLLL                           _     L
808 PRINT"TTL          LLLLLLLLLLLLLLLLL    LLL    L
810 PRINT"TTLLLLLL                     LLL        L
812 PRINT"TTL                    LLL _            L
814 PRINT"TTL   LLLLLL     LLLLLL  _              L
816 PRINT"TTL       _                             L
818 PRINT"TTL      LLLLLL     LLLLL    LLLLLLLLLLLL L
820 PRINT"TTL                                     L"
822 PRINT"TTL            LLLLLLLLLLLLLLLLLLLL _   L"
824 PRINT"TTL     LLL        _                    L"
826 PRINT"TTL                                     L
828 PRINT"TTL    LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL
830 PRINT"TTL                                     L"
832 PRINT"TTLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL     L
834 PRINT"TTL     _         _       _             L"
836 PRINT"TTLLLLLLL   LLLLLLL   LLLLLLL   LLLLLL   L"
838 PRINT"TTL                                     L"
840 PRINT"TTL    LLLLLL   LLLLLLLL   LLLLLL   LLL L
842 PRINT"TTL                                    _L
844 PRINT"TTLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL"
845 PRINT"[graphics]            THE FAIRGROUND        "
849 GOSUB63000
850 GOSUB4000
851 CC=8
860 PRINT"[graphics]   LLLLLLLLLLLLLLLLLLLLLLLLLL";
```





Program listing

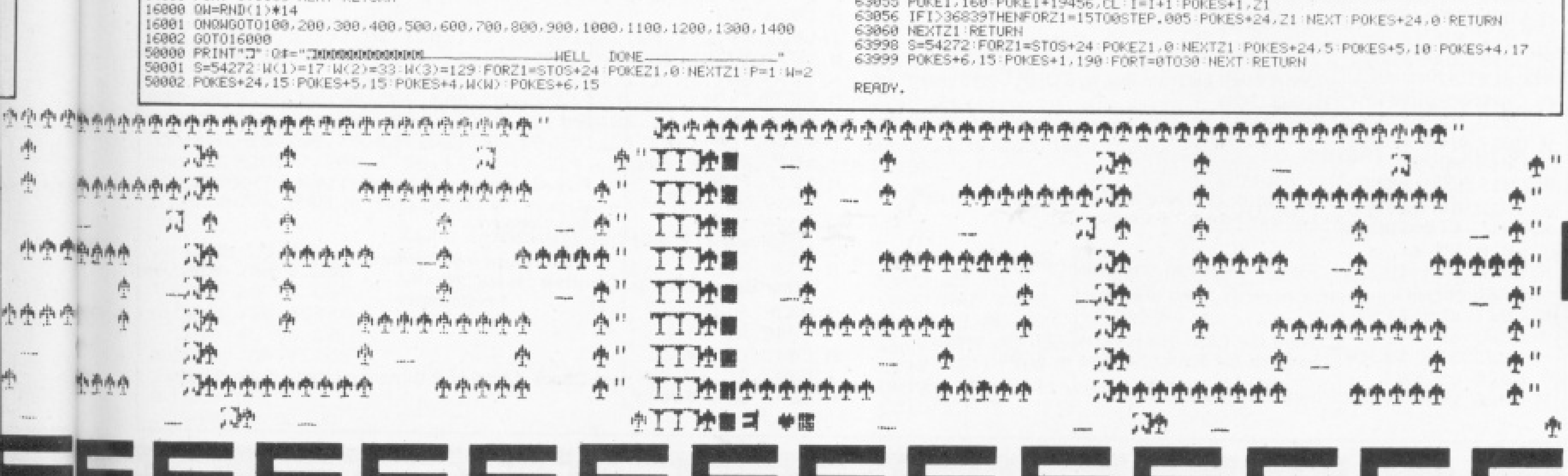
```
861 PRINT"■■■■■⊠⊠⊠⊠⊠◣◥L◥L◥L◥L◥LL";:PRINT"▶▶▶▶▶▶▶▶▶▶▶▶▶▶⊠⊠⊠⊠◣◥L◥L◥L◥L◥L◥L◥LL"
872 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
874 IFA1=69ORA2=69ORA3=69THENGOSUB63998:SC=SC+25:CA=CA+1
876 IFPEEK(A+M+40)=73ORPEEK(A+1+M+40)=73ORPEEK(A+2+M+40)=73THENGOSUB4400
880 IFA1=83ANDCA>14THENGOSUB50000:GOSUB5000:GOTO16000
895 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO900
896 IFA1=83ANDCA>14ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
897 PRINT"■⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠■    ◥L◥L◥L◥L◥L◥L◥L◥L◥L◥L◥L◥LL";
898 PRINT"■■■■■⊠⊠⊠⊠⊠◥L◥L◥L◥L◥L◥L";:PRINT"▶▶▶▶▶▶▶▶▶▶▶▶▶▶⊠⊠⊠⊠◥L◥L◥L◥L◥L◥L◥L◥L"
899 GOTO850
900 POKE53280,8:POKE53281,9:PRINT"⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠":CA=0:WA=9:M=0:QQ=85
902 PRINT"╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱"
904 PRINT"TTI╱╱        ╱         ╱                          ╱"
906 PRINT"TTI╱╱‾   ╱    ╱    ╱╱    ╱     ╱    ╱╱╱╱╱╱╱╱╱    ╱"
908 PRINT"TTI╱╱    ╱    ╱    ╱      ╱╱╱╱╱╱               ╱   ╱"
910 PRINT"TTI╱╱    ╱ ‾  ╱    ╱‾          ╱╱╱╱╱╱╱╱╱╱    ╱   ╱"
912 PRINT"TTI╱╱    ╱    ╱    ╱╱╱╱╱╱    ╱╱              ╱   ╱"
914 PRINT"TTI╱╱    ╱    ╱          ╱    ╱╱    ╱╱╱╱╱╱╱╱╱    ╱"
916 PRINT"TTI╱╱    ╱    ╱╱╱╱╱╱    ╱                  ╱    ╱"
918 PRINT"TTI╱‾    ╱               ╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱      ╱"
920 PRINT"TTI╱╱    ╱╱╱╱╱╱╱         ╱                       ╱"
922 PRINT"TTI╱‾         ╱╱╱   ╱    ╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱"
924 PRINT"TTI╱╱╱╱╱╱╱╱╱        ╱╱╱╱                      ╱"
926 PRINT"TTI╱‾       ╱╱╱        ╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱    ╱"
928 PRINT"TTI╱╱    ╱    ╱╱╱          ╱╱╱■♦■              ╱"
930 PRINT"TTI╱╱    ╱          ╱╱╱       ╱╱╱╱              ╱"
932 PRINT"TTI╱╱    ╱  ╱╱╱╱         ╱╱╱       ╱╱╱╱╱╱╱╱╱╱╱╱╱╱"
934 PRINT"TTI╱‾    ╱      ╱╱╱       ╱╱╱                   ╱"
936 PRINT"TTI╱╱   ╱╱╱╱     ╱╱╱        ╱╱╱╱╱╱╱╱╱╱╱╱╱      ╱"
938 PRINT"TTI╱╱    ╱‾                                     ╱"
940 PRINT"TTI╱╲    ╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱"
942 PRINT"TTI╱╲‾                                          ╱"
944 PRINT"TTI╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱"
945 PRINT"⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠ ■              ■THE MAZE 2■          ■"
949 GOSUB63000
950 GOSUB4000
951 CC=8
966 PRINT"■⊠⊠⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥⊠⊠◥"
967 PRINT"■⊠⊠⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱⊠⊠╱"
972 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
974 IFA1=69ORA2=69ORA3=69THENGOSUB63998:SC=SC+25:CA=CA+1
975 IFA1=77ORA2=77ORA3=77THEN60000
976 IFA1=78ORA2=78ORA3=78THEN60000
995 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO1000
997 IFA1=83ANDCA>17ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
998 IFA1=83ANDCA>17THENGOSUB50000:GOSUB5000:GOTO16000
999 GOTO950
1000 POKE53280,5:POKE53281,0:PRINT"⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠":M=0:CA=0:WA=10:QQ=65
1002 PRINT"■♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
1004 PRINT"TT♣■♦■          ♣      ♣♣♣♣♣♣♣♣♣♣      ♣     ♣"
1006 PRINT"TT♣♣♣♣♣♣♣♣                                   ♣"
1008 PRINT"TT♣         ♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣   ♣"
1010 PRINT"TT♣♣♣♣          ‾                            ♣"
1012 PRINT"TT♣             ♣♣♣♣♣♣♣       ♣♣♣♣♣♣         ♣"
1014 PRINT"TT♣    ♣♣♣♣            ‾♣♣♣♣          ‾♣♣♣♣♣♣♣"
1016 PRINT"TT♣                                          ♣"
1018 PRINT"TT♣   ♣♣♣♣♣♣♣ ‾  ♣♣♣♣♣♣♣♣♣♣      ‾♣♣♣♣♣♣   ♣"
1020 PRINT"TT♣                                          ♣"
1022 PRINT"TT♣         ♣♣♣♣♣♣♣♣♣♣       ♣♣♣♣♣♣♣♣♣      ♣"
1024 PRINT"TT♣                    ‾                     ♣"
1026 PRINT"TT♣♣♣♣♣♣♣♣         ♣♣♣♣♣♣♣♣        ♣♣♣♣♣   ♣"
1028 PRINT"TT♣                                     ‾    ♣"
1030 PRINT"TT♣      ♣♣♣♣♣♣♣♣       ♣♣♣♣♣♣♣♣♣    ♣♣♣♣♣♣♣♣"
1032 PRINT"TT♣         ‾                                ♣"
1034 PRINT"TT♣♣♣♣         ♣♣♣♣♣♣♣♣♣♣♣♣♣     ‾ ♣♣♣♣♣♣♣♣♣♣"
1036 PRINT"TT♣            ‾           ‾                 ♣"
1038 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣ ‾ ♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
1040 PRINT"TT♣        ‾            ♣     ♣              ♣"
1042 PRINT"TT♣ ‾ ♣♣♣   ♣♣♣ ‾                ♣♣♣ ‾ ♣♣   ♣"
1044 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
1045 PRINT"⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠ ■              ■THE DARK ROOM 2■          ■"
1049 GOSUB63000
1050 GOSUB4000
1072 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
1074 IFA1=82ORA2=82ORA3=82THENGOSUB63998:SC=SC+50:CA=CA+1
1076 POKEA+M+40+CO,9:POKEA+1+M+40+CO,9:POKEA+2+M+40+CO,9
1078 POKEA+M-40+CO,9:POKEA+1+M-40+CO,9:POKEA+2+M-40+CO,9
1080 POKEA+M-1+CO,9:POKEA+2+M+1+CO,9
1095 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO1100
1097 IFA1=83ANDCA>15ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
1098 IFA1=83ANDCA>15THENGOSUB50000:GOSUB5000:GOTO1100
1099 GOTO1050
1100 POKE53280,9:POKE53281,0:PRINT"⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠":WA=11:CA=0:QQ=65
1101 M=0:M=741:DI=0
1102 PRINT"♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
1104 PRINT"TT♣                ■TTT■        ‾    ‾          ♣"
1106 PRINT"TT♣♣♣♣♣♣♣♣     ♣■╱╱╱╱╱╱╱╱╱■■╱╱╱╱╱╱♣    ♣♣♣♣♣♣♣♣♣♣"
1108 PRINT"TT♣     ‾      ♣♣♣♣■╱╱╱■■╱╱╱╱♣♣♣♣    ♣♣        ♣"
1110 PRINT"TT♣♣♣♣♣♣♣♣     ♣■╱■■╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱♣    ♣   ‾   ♣"
1112 PRINT"TT♣       ‾    ♣■╱╱╱╱╱♣♣♣♣■╱╱■■╱╱╱♣     ♣    ♣♣♣♣♣"
1114 PRINT"TT♣    ♣♣♣♣♣♣♣♣♣■╱╱╱╱╱╱╱╱╱╱╱■■╱╱╱╱♣             ♣"
1116 PRINT"TT♣       ‾    ♣■╱╱♣♣♣♣■╱■■╱╱╱♣♣♣♣■╱╱♣    ♣♣♣♣♣    ♣"
1118 PRINT"TT♣♣♣♣♣♣♣♣     ♣■╱╱╱╱╱■■╱╱╱╱╱╱╱╱╱╱╱♣    ♣    ‾   ♣"
1120 PRINT"TT♣       ‾    ♣■╱╱╱╱╱╱╱♣♣■╱╱╱╱■■╱╱♣     ♣    ♣♣♣♣♣"
1122 PRINT"TT♣    ♣♣♣♣♣♣♣♣♣■╱╱╱■■╱╱╱╱╱╱╱╱╱╱╱╱╱♣    ♣‾        ♣"
1124 PRINT"TT♣       ‾    ♣♣♣■╱╱╱╱╱■■╱╱╱╱╱♣♣♣    ♣♣♣♣♣     ♣"
1126 PRINT"TT♣♣♣♣♣♣♣♣     ♣■╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱♣ ■♦■ ♣     ‾   ♣"
1128 PRINT"TT♣       ‾    ♣♣■╱╱╱■■╱╱╱■■╱╱╱■■╱╱♣♣     ♣     ♣♣♣♣♣"
1130 PRINT"TT♣    ♣♣♣♣♣♣♣♣♣■╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱♣     ♣         ♣"
1132 PRINT"TT♣     ‾  ■   ♣■╱╱♣♣♣■╱╱╱■■■╱╱╱♣♣■╱╱♣ ‾ ♣♣♣♣♣       ♣"
1134 PRINT"TT♣♣♣♣♣♣♣♣     ♣■╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱♣    ♣    ‾   ♣"
1136 PRINT"TT♣   ‾        ♣■╱╱■■╱╱╱♣♣♣♣■╱╱╱╱╱♣     ♣    ♣♣♣♣♣"
1138 PRINT"TT♣    ♣♣♣♣♣♣♣♣♣■╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱╱♣     ♣         ♣"
1140 PRINT"TT♣          ‾   ♣♣♣♣■╱╱╱╱╱╱╱╱╱♣♣♣♣    ♣♣♣♣ ‾ ♣♣"
1142 PRINT"TT♣              ‾                  ♣         ‾  ♣"
1144 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
1145 PRINT"⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠ ■              ■FALLING DOWN■"
1149 GOSUB63000
1150 GOSUB4000
1172 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
1174 IFA1=82ORA2=82ORA3=82THENGOSUB63998:SC=SC+50:CA=CA+1
1176 IFA1=79ORA2=79ORA3=79THEN60000
1195 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO1200
1197 IFA2=83ANDCA>17ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
1198 IFA2=83ANDCA>17THENGOSUB50000:GOSUB5000:GOTO1200
1199 GOTO1150
1200 POKE53280,4:POKE53281,0:PRINT"⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠":WA=12:QQ=65:CA=0
1201 M=0:DI=0:CC=4
1202 PRINT"♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
1204 PRINT"TT♣■   ‾     ♣               ■♣    ♣    ‾      ■      ♣"
1206 PRINT"TT♣■     ♣ ‾ ♣    ♣♣♣♣♣♣♣■♣    ♣    ♣♣♣♣♣♣♣♣♣     ♣"
1208 PRINT"TT♣■     ♣         ‾       ■ ♣    ♣          ♣      ‾ ♣"
1210 PRINT"TT♣■     ♣    ♣♣♣♣♣♣♣♣     ■♣    ♣♣♣♣♣   ‾♣     ♣♣♣♣♣"
1212 PRINT"TT♣■   ‾♣               ♣  ‾■♣    ♣          ♣      ‾ ♣"
1214 PRINT"TT♣■     ♣♣♣♣♣♣♣♣     ♣    ■♣    ♣    ♣♣♣♣♣♣♣♣♣     ♣"
1216 PRINT"TT♣■           ‾   ♣       ■♣          ♣ ‾       ♣    ♣"
1218 PRINT"TT♣♣♣♣♣♣♣♣♣♣    ♣♣♣♣♣      ■♣♣♣♣♣♣♣♣♣     ♣♣♣♣♣     ♣"
1220 PRINT"TT♣■■ ♦■      ‾              ‾  ■♣   ‾                ♣"
1222 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
1224 PRINT"TT♣             ‾             ♣               ‾    ♣"
1226 PRINT"TT♣■ ‾ ♣♣♣♣     ♣♣♣♣♣      ■♣♣♣    ♣♣♣    ♣♣♣    ♣♣♣"
1228 PRINT"TT♣                           ♣               ‾    ♣"
1230 PRINT"TT♣■♣♣♣♣♣♣♣♣♣♣♣♣ ‾ ♣♣♣♣■♣♣    ♣♣♣    ♣♣♣    ♣♣♣♣"
1232 PRINT"TT♣                           ♣                    ♣"
1234 PRINT"TT♣■    ♣♣♣♣♣    ‾♣♣♣♣    ■ ♣   ♣♣♣ ‾ ♣♣♣    ♣♣♣  ♣"
1236 PRINT"TT♣                           ♣                    ♣"
1238 PRINT"TT♣■♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣ ‾ ■♣ ♣♣♣    ♣♣♣ ‾ ♣♣♣    ♣"
1240 PRINT"TT♣■               ♣          ■ ♣          ‾ ♣       ♣"
1242 PRINT"TT♣     ♣           ♣ ‾     ♣         ♣          ♣ ‾♣"
1244 PRINT"TT♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣♣"
1245 PRINT"⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠ ■              ■ALL CHANGE■          ■"
1249 GOSUB63000
1250 GOSUB4000
1251 CC=4
1272 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
1274 IFA1=85ORA2=85ORA3=85THENGOSUB14000
1275 IFA1=82ORA2=82ORA3=82THENGOSUB63998:SC=SC+50:CA=CA+1
1276 POKEA+M+40+CO,8:POKEA+1+M+40+CO,8:POKEA+2+M+40+CO,8
1278 POKEA+M-40+CO,8:POKEA+1+M-40+CO,8:POKEA+2+M-40+CO,8
1280 POKEA+M-1+CO,8:POKEA+2+M+1+CO,8
1282 IFCA=6THENM=760:CA=7
1284 IFCA=12THENM=-10:CA=13
1286 IFCA=18THENM=741:CA=19
1295 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO1300
1297 IFA1=83ANDCA>25ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
1298 IFA1=83ANDCA>25THENGOSUB50000:GOSUB5000:GOTO1100
1299 GOTO1250
1300 POKE53280,5:POKE53281,11:PRINT"⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠":WA=13:CA=0:QQ=95
1301 M=0:M=M-2
1302 PRINT"◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥"
1304 PRINT"TTI◥■♦■                                          ◥"
1306 PRINT"TTI◥◥◥◥◥    ◥    ◥◥◥   ◥◥◥◥◥◥    ◥◥    ◥    ◥◥◥◥◥"
1308 PRINT"TTI◥   ‾     ◥                ‾          ◥         ◥"
1310 PRINT"TTI◥    ◥    ◥    ◥◥◥◥◥◥    ◥◥◥◥◥◥◥    ◥    ◥    ◥"
1312 PRINT"TTI◥    ◥              ◥    ◥    ‾        ◥         ◥"
1314 PRINT"TTI◥    ◥    ◥    ◥◥   ◥    ◥    ◥◥       ◥    ◥    ◥"
1316 PRINT"TTI◥    ◥    ◥    ◥◥ ‾ ◥    ◥    ◥◥       ◥    ◥    ◥"
1318 PRINT"TTI◥‾             ◥◥   ◥    ◥    ◥◥                 ◥"
1320 PRINT"TTI◥    ◥◥◥◥◥                           ◥◥◥◥◥      ◥"
1322 PRINT"TTI◥         ◥◥◥◥◥◥ ‾ ◥◥◥    ◥◥◥◥◥◥◥            ◥"
1324 PRINT"TTI◥◥◥◥◥◥◥                                ◥◥◥◥◥◥"
1326 PRINT"TTI◥          ◥◥◥◥◥   ◥◥◥◥◥◥◥    ◥◥◥◥◥         ◥"
1328 PRINT"TTI◥◥◥◥◥◥◥◥◥◥ ‾                         ◥◥◥◥◥◥◥◥◥"
1330 PRINT"TTI◥              ◥◥◥◥◥◥    ◥◥◥◥◥◥              ◥"
1332 PRINT"TTI◥    ◥    ◥                  ‾        ◥    ◥    ◥"
1334 PRINT"TTI◥    ◥                  ◥                  ◥    +"
1336 PRINT"TTI◥    ◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥    ◥"
1338 PRINT"TTI◥    ◥         ‾                           ◥    ◥"
1340 PRINT"TTI◥    ◥    ◥◥◥    ◥    ◥◥◥    ◥    ◥◥◥    ◥    ◥"
1342 PRINT"TTI◥                                               ◥"
1344 PRINT"TTI◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥◥"
1345 PRINT"⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠ ■              ■LITTER BUG■          ■"
1349 GOSUB63000
1350 GOSUB4000
1352 JP=RND(1)*5:ONJPGOSUB2000,2010,2020,2030
1372 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
1374 IFA1=82ORA2=82ORA3=82THENGOSUB63998:SC=SC+50:CA=CA+1
1376 IFA1=PJORA2=PJORA3=PJTHEN60000
1395 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:GOTO1400
1397 IFA1=83ANDCA>8ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
1398 IFA1=83ANDCA>8THENGOSUB50000:GOSUB5000:GOTO1400
1399 GOTO1350
1400 POKE53280,8:POKE53281,15:PRINT"⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠⊠"
1401 CA=0:WA=14:QQ=90:M=0:M=M+15
1402 PRINT"■♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦"
1404 PRINT"TT♦×××××××××××××××××××××××××■_■××××××××××××××××♦"
1406 PRINT"TT♦♦♦ ◥◥◥♦♦♦♦♦♦♦♦♦♦♦×■_■××××××××♦♦♦♦♦♦♦♦♦♦♦◥◥◥♦♦♦"
1408 PRINT"TT♦     ■_■              ♦××××××××××♦ ■_■          ♦"
1410 PRINT"TT♦♦♦♦♦♦♦♦♦♦♦◥◥◥♦♦××××××××××♦◥◥◥♦♦♦♦♦♦♦♦♦♦♦♦♦"
1412 PRINT"TT♦            ■_■     ♦××××××××××♦              ■_■ ♦"
1414 PRINT"TT♦♦◥◥◥♦♦♦♦♦♦♦♦♦♦♦♦××××××××××♦♦♦♦♦♦♦♦♦♦♦◥◥◥♦♦♦"
1416 PRINT"TT♦                ■_■ ♦×××××■_■××××♦              ♦"
1418 PRINT"TT♦♦♦♦♦♦♦♦♦♦◥◥◥♦♦♦ ◥◥◥◥◥■■◥◥◥♦♦ ◥◥◥♦♦♦♦♦♦♦♦♦♦♦♦"
1420 PRINT"TT♦          ■_■        ♦×××××■_■××××♦         ■_■        ♦"
1422 PRINT"TT♦♦◥◥◥♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦◥◥◥♦♦♦♦♦♦♦♦♦♦♦♦♦♦◥◥◥♦♦♦"
1424 PRINT"TT♦           ■_         _■                        ♦"
```



Program listing

```
1426 PRINT"TT♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠▼▼▼♠"
1428 PRINT"TT♠xxxxxxxxxxxxxxxx          xxxxxxxxxxxxx■_i♠"
1430 PRINT"TT♠xxxxxxxxxxx■_i♠♠♠♠♠♠♠▼▼▼♠xxxx■_ixxxxxxxxx♠"
1432 PRINT"TT♠xxxxx■_ixxxxxxxx♠          ♠xxxxxxxxxxxxxxx♠"
1434 PRINT"TT♠▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼▼♠"
1436 PRINT"TT♠xxxxxxxxxxxxxxxxxxxx■_ixxxxxxxxxxxxxxxxxx♠"
1438 PRINT"TT♠▼▼▼▼▼▼▼♠♠♠♠♠♠♠♠ ▼▼▼▼▼▼▼▼▼▼♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠"
1440 PRINT"TT♠xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx♠"
1442 PRINT"TT♠      ■_i                                ♠"
1444 PRINT"TT♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠"
1445 PRINT"■■■■■■■■■■■■■■■■■■■■■■■■■ ■           ■LITTER BUG 2■
1449 GOSUB63000
1450 GOSUB4000
1472 A1=PEEK(A+M):A2=PEEK(A+1+M):A3=PEEK(A+2+M)
1474 IFA1=82ORA2=82ORA3=82THENGOSUB63998:SC=SC+50:CA=CA+1
1476 IFA1=86ORA2=86ORA3=86THENCC=CC+1:IFCC>14THENCC=0
1478 IFA1=86ORA2=86ORA3=86THENPJ=90
1495 IFAA>500THENVE=VE+2:IFVE>5THENGOSUB6000:VE=0:AA=0:GOTO55000
1497 IFA2=83ANDCA>13ANDPL=1THENGOSUB50000:GOSUB5000:GOTO16000
1498 IFA2=83ANDCA>13THENGOSUB50000:GOSUB5000:GOTO100
1499 GOTO1450
1500 GOTO1500
2000 CC=1:PJ=91:RETURN
2010 CC=0:PJ=92:RETURN
2020 CC=7:PJ=93:RETURN
2030 CC=8:PJ=94:RETURN
2040 RETURN
4000 POKEA+M,X:POKEA+M+CO,CC:POKEA+1+M,Y:POKEA+1+M+CO,CC:POKEA+2+M,Z:POKEA+2+M+C
O,CC
4002 POKEAIR+MM,32:MM=MM-.05:BO=BO-1:IFAIR+MM<36800THEN60000
4003 IFUP=1THENPU=PU+1:IFPU>7THENHA=0:PU=0
4005 K=PEEK(56320)
4006 KE=PEEK(197):IFKE=1THEN55000
4010 IFK=119THENGOSUB4200
4015 IFK=123THENGOSUB4400
4020 IFK<>111ANDFB<1THENGOSUB4600
4021 IFK=111THENFB=1:GOSUB8800
4025 IFK=126THENFB=1:GOSUB4800
4026 IFK<>126THENFB=0
4100 PRINT"■■ SCORE";SC:PRINTSPC(15)"TLIVES ";LI:PRINTSPC(28)"THIGH";HI
4110 IFSC>HITHENLETHI=SC
4199 RETURN
4200 IFPEEK(A+M+1)=0ORPEEK(A+1+M+1)=0ORPEEK(A+2+M+1)=0THEN60000
4300 POKEA+M,PJ:M=M+1:X=70:Y=71:Z=72:DI=1
4399 RETURN
4400 IFPEEK(A+M-1)=0ORPEEK(A+1+M-1)=0ORPEEK(A+2+M-1)=0THEN60000
4500 POKEA+2+M,PJ:M=M-1:X=66:Y=67:Z=68:DI=0
4599 RETURN
4600 IFPEEK(A+M+40)=0ORPEEK(A+1+M+40)=0ORPEEK(A+2+M+40)=0THENDR=0:H=0:CX=0:RE
TURN
4700 POKEA+2+M,PJ:POKEA+1+M,PJ:POKEA+M,PJ:M=M+40:DR=DR+1:IFDR>12THEN60000
4750 IFDI=0ANDUP<1THENX=66:Y=67:Z=68
4751 IFDI=1ANDUP<1THENX=70:Y=71:Z=72
4755 UP=0
4799 RETURN
4800 IFHA=1THENRETURN
4801 IFPEEK(A+M-40)=0ORPEEK(A+1+M-40)=0ORPEEK(A+2+M-40)=0THEN60000
4900 POKEA+2+M,PJ:POKEA+1+M,PJ:POKEA+M,PJ:M=M-40
4952 IFDI=0ANDUP=0THENX=66:Y=67:Z=68
4953 IFDI=1ANDUP=0THENX=70:Y=71:Z=72
4999 RETURN
5000 PRINT53280,10:POKE53281,14:PRINT"■■■■■■■■■■■■■"
5002 PRINT"■♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠"
5004 PRINT"■♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠■BONUS!■♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠"
5005 PRINT"■♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠"
5006 FORT=0TOBOSTEP11:SC=SC+11
5008 PRINT"■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■";:PRINTSC;
5010 NEXT
5020 S=54272:POKES+24,15
5021 POKES,0:POKES+1,0:POKES+2,0:POKES+3,0:POKES+4,0:POKES+5,0:POKES+6,0
5025 POKES+5,32:POKES+6,240:POKES+0,17:POKES+4,17:POKE54296,15
5030 FORZ1=255TO100STEP-1:POKES+1,Z1:POKE53280,Z1:NEXTZ1
5040 POKES+4,0:POKES+5,0:POKES+6,0
5060 RETURN
6000 S=54272:FORZ1=0TO15STEP7.5:POKES+24,Z1:NEXT
6005 POKES+5,17:POKES+6,248:POKES+0,137:POKES+4,33
6010 FORZ1=65TO10STEP-2:POKES+1,Z1:POKE53281,Z1:NEXTZ1:FORZ1=15TO0STEP-7.5
6030 POKES+24,Z1:NEXT:RETURN
7000 S=54272:POKES+24,10
7001 POKES+4,0:POKES+5,0:POKES+6,0
7005 POKES+5,17:POKES+6,248:POKES,137:POKES+4,17
7010 POKES+1,Z2:Z2=Z2-25:IFZ2<5THENZ2=150
7030 RETURN
8800 IFCX>0THENRETURN
8801 IFPEEK(A+M-40)=0ORPEEK(A+1+M-40)=0ORPEEK(A+2+M-40)=0THEN60000
8900 POKEA+2+M,PJ:POKEA+1+M,PJ:POKEA+M,PJ:M=M-40:CX=CX+1:FB=0
8952 IFDI=0ANDUP=0THENX=66:Y=67:Z=68
8953 IFDI=1ANDUP=0THENX=70:Y=71:Z=72
8999 RETURN
12000 IFPEEK(A+M+40)=73ORPEEK(A+1+M+40)=73ORPEEK(A+2+M+40)=73THENM=M-1
12001 IFPEEK(A+M+40)=73ORPEEK(A+1+M+40)=73ORPEEK(A+2+M+40)=73THENM=M-1
12002 RETURN
13001 S=54272:POKES+24,15:POKES+5,190:POKES+6,240:POKES,7:POKES+4,17
13005 FORTT=20TO0STEP-20:POKE53281,TT:POKES+1,TT:NEXTTT
13006 GOTO13000
14000 FORT=0TO100:NEXT:RETURN
16000 QW=RND(1)*14
16001 ONQWGOTO100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,1400
16002 GOTO16000
50000 PRINT"■":O$="■■■■■■■■■■■■■■___________________WELL  DONE_______________"
50001 S=54272:W(1)=17:W(2)=33:W(3)=129:FORZ1=STOS+24:POKEZ1,0:NEXTZ1:P=1:W=2
50002 POKES+24,15:POKES+5,15:POKES+4,W(W):POKES+6,15
50005 FORZ1=1TO250STEP10:POKES+1,Z1:POKES+1,255-Z1:PRINTO$:POKE646,INT(RND(1)*15
50006 NEXTZ1:RETURN
51000 Z2=0:Z=0:S=0:PRINT"■":POKE53280,11:POKE53281,0:FORT=0TO50
51010 PRINT"♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠GAME  OVER♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠"
51015 POKE646,INT(RND(1)*15):NEXT:GOTO55000
55000 CLR:POKE53280,8:POKE53281,8
55001 PRINT"■■■■■■■■■■■■■■■■■■■■■■////////////////////////////////////////"
55002 PRINT"TT/■ PLEASE WAIT TO VEIW FOURTEEN SCREENS ■/
55003 PRINT"TT/ ■    _____PRESS ■[F.1]■ TO PLAY._____ ■   /
55004 PRINT"TT/     _____PRESS [R] FOR RANDOM_____     /
55005 PRINT"TT/ ■♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠■ /
55006 PRINT"TT/                                      /"
55007 PRINT"TT/   ___________BY F.G.TOUT___________  /
55008 PRINT"TT/  HIT RETURN FOR TITLE PAGE ANY TIME  /
55009 PRINT"TT/ ■♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠♠■ /
55010 PRINT"TT/
55011 PRINT"TT/
55012 PRINT"TT/
55013 PRINT"TT/
55014 PRINT"TT/
55015 PRINT"TT/
55016 PRINT"TT/
55017 PRINT"TT/
55018 PRINT"TT/
55019 PRINT"TT/
55020 PRINT"TT/
55022 PRINT"TT/
55023 PRINT"TT/
55065 PRINT"TT////////////////////////////////////////"
55900 GETA$
55905 IFAA<26THEN59000
55910 AA=AA+10:IFAA>500THEN100
55915 IFA$=""THEN55900
55950 IFA$="■"THEN10
55955 IFA$="R"THENPL=1:LI=3:GOSUB56000
55999 GOTO55900
56000 QW=RND(1)*10:ONQWGOTO100,200,300,400,500,600,700,800,900,1000,1100
56010 RETURN
57000 S=54272:POKES+4,0:POKES+5,0:POKES+6,0
57010 FORZ1=0TO15STEP.5:POKES+24,Z1:NEXT:POKES+5,17:POKES+6,248
57015 POKES,137:POKES+4,33:FORTT=0TO25:FORZ2=100TO115:POKES+1,Z2
57020 NEXTZ2:NEXTTT
57030 END
57050 GOTO57015
59000 I1=54272:I2=54279:I3=54286:POKE54279+4,0:POKE54296,10:POKE54296,15
59001 POKE54279+1,0:POKE54279+2,0:POKE54279+3,0:POKE54279+5,0
59005 H1=I1+1:H2=I1+1:H3=I3+1:V1=I1+4:V2=I2+4:V3=I3+4:POKE54296,15
59010 POKEV1+1,30:POKEV1+2,17:POKEV2+1,40:POKEV2+2,32:POKEV3+1,40:POKEV3+2,240
59020 T=T+1:POKEV1,126:POKEV2,140:POKEV3,32
59030 READS:IFS=0THEN59081
59040 READX1,Y1,X2,Y2,X3,Y2
59050 IFX1THENPOKEH1,X1:POKEI1,Y1:POKEV1,33
59060 IFX2THENPOKEH2,X2:POKEI2,Y2:POKEV2,17
59065 IFX3THENPOKEH3,X3:POKEI3,Y3:POKEV3,33
59070 T=T+S
59072 IFT>TITHEN59072
59075 GOTO59020
59081 AA=27:GOTO55900
59082 DATA15,17,37,68,149,4,73,15,21,154,0,0,0,0,15,25,177,0,0,6,108,15,28,214
59083 DATA0,0,0,0
59084 DATA15,30,141,115,88,7,163,15,28,214,0,0,0,0,15,25,177,0,0,6,108,15,21,154
59085 DATA0,0,0,0,15,17,37,68,149,4,73,15,21,154,0,0,0,0,15,25,177,0,0,6,108
59086 DATA15,28,214,0,0,0,0,15,30,141,115,68,7,163,15,28,214,0,0,0,0,15,25,177
59087 DATA0,0,6,108,15,21,154,0,0,0,0,15,22,227,91,140,5,185,15,28,214,0,0,0,0
59088 DATA15,34,75,0,0,8,147,15,38,126,0,0,0,0,15,40,200,163,31,10,60,15,38,126
59089 DATA0,0,0,0,15,34,75,0,0,8,147,15,28,214,0,0,0,0,15,17,37,68,149,4,73
59090 DATA15,21,154,0,0,0,0,15,25,177,0,0,6,108,15,28,214,0,0,0,0,15,30,141
59091 DATA122,52,7,163,15,28,214,0,0,0,0,15,25,177,0,0,6,108,15,21,154,0,0,0,0
59092 DATA25,17,37,68,149,4,73,20,0,0,0,0,0,0
59184 DATA0
60000 TA=A+M:CO=19456:Z1=0:Z2=0:S=0:W=0
60001 S=54272:W(1)=17:W(2)=33:W(3)=129:FORZ1=STOS+24:POKEZ1,0:NEXTZ1:P=1:W=3
60002 POKES+24,15:POKES+5,15:POKES+4,W(W):POKES+6,15
60006 FORZ1=20TO100STEP20:FORZ2=1TO5
60009 POKES+1,Z1-Z2:POKES+1,Z2+50
60010 POKETA,87:POKETA+CO,2:POKETA+1,87:POKETA+1+CO,2:POKETA+2,87:POKETA+2+CO,2
60011 POKETA,88:POKETA+CO,2:POKETA+1,88:POKETA+1+CO,2:POKETA+2,88:POKETA+2+CO,2
60012 POKETA,89:POKETA+CO,2:POKETA+1,89:POKETA+1+CO,2:POKETA+2,89:POKETA+2+CO,2
60015 NEXTZ2,Z1:Z1=0:Z2=0
60020 POKETA-40,81:POKETA-40+CO,1:POKETA-39,81:POKETA-39+CO,1
60025 POKETA-38,81:POKETA-38+CO,1
60030 FORT=0TO1000:NEXT
60035 S=54272:W(1)=17:FORZ1=STOS+24:POKEZ1,0:NEXTZ1:P=1:W=2
60036 POKES+24,15:POKES+5,15:POKES+4,33:POKES+6,15
60040 W=2:FORZ1=75TO5STEP-1:POKES+1,Z1:POKES,Z1:POKE53281,1:POKE53281,0:NEXTZ1
60100 M=0:LI=LI-1:IFLI<1THEN51000
60150 POKES+24,10:POKES+24,5:POKES+24,1
60151 ONWAGOTO100,200,300,400,500,600,700,800,900,1000,1100,1200,1300
60199 JJ=0:GOTO60151
63000 A=35940:CO=19456:CC=8:X=66:Y=67:Z=68:AIR=36840:CL=1:MM=0:BO=500:PJ=32
63001 IFAA>500THENRETURN
63002 S=54272:POKES+24,15
63003 POKES+4,0:POKES+5,0:POKES+6,0:POKES+1,0:POKES,0:POKES+24,15
63004 I=36800
63005 POKES+24,15:POKES+5,190:POKES+4,17:POKES+6,15:FORZ1=17TO97STEP2
63055 POKEI,160:POKEI+19456,CL:I=I+1:POKES+1,Z1
63056 IFI>36839THENFORZ1=15TO0STEP.005:POKES+24,Z1:NEXT:POKES+24,0:RETURN
63060 NEXTZ1:RETURN
63998 S=54272:FORZ1=STOS+24:POKEZ1,0:NEXTZ1:POKES+24,5:POKES+5,10:POKES+4,17
63999 POKES+6,15:POKES+1,190:FORT=0TO30:NEXT:RETURN

READY.
```



Our man in Eire, John McHale, makes some sense of the intricacies of raster interrupts.

RASTER INTERRUPTS

This article consists of four demonstration programs on raster graphics. They are as follows:

1. Splitting screen colour
2. Mixing user-defined graphics with standard 'Pet' graphics.
3. Switching between hi-resolution and text.
4. Displaying more than 8 'sprites' on screen.

The program listings should be typed in and saved immediately before any attempt is made to run them. If the programs have been typed in correctly, you will be given the option of running the demonstration, otherwise error reports will be printed on screen.

One warning: before even attempting to understand the concept of raster interrupts, you should have a fairly reasonable understanding of machine language.

Program 1 Listing

```
10 REM ********************************
20 REM * RASTER DEMONSTRATION PROG#1 *
30 REM *                              *
40 REM * SPLITTING SCREEN COLOUR      *
50 REM ********************************
60 TL=0:SA=49152:C=0
70 READA:IFA=-1THEN100
80 POKESA+C,A:C=C+1:TL=TL+A
90 GOTO70
100 PRINT"";:REM * SHIFT & CLR/HOME *
110 IFC=134THEN150
120 IFC<134THENPRINT"TOO FEW ";:GOTO140
130 PRINT"TOO MANY ";
140 PRINT"DATA ITEMS":REM * CU DOWN *
150 IFTL=14874THEN180
160 PRINT"ERROR IN DATA INPUT"
170 STOP
180 PRINT"PRESS ANY KEY TO SEE DEMONSTRATION."
190 PRINT"PRESS 'CURSOR UP/DOWN' WITH OR WITHOUT  ":REM * CU DOWN X 2 *
200 PRINT"'SHIFT', TO MOVE THE SPLIT UP OR DOWN   "
210 PRINT"THE SCREEN."
220 POKE198,0:WAIT198,1
230 SYS49152
240 DATA32,9,192,32,111,192,76,3,192,120,173,14,220,41,254,141,14,220,173,17
250 DATA208,41,127,141,17,208,169,144,141,18,208,133,2,169,64,141,20,3,169,192
260 DATA141,21,3,169,147,32,210,255,173,26,208,9,1,141,26,208,88,96,104,170,104
270 DATA168,104,64,169,1,44,25,208,240,243,173,33,208,41,15,201,6,208,21,169,1
280 DATA141,33,208,169,0,141,18,208,173,25,208,9,1,141,25,208,76,58,192,169,6
290 DATA141,33,208,165,2,76,87,192,32,159,255,165,197,201,7,240,1,96,173,141,2
300 DATA41,1,208,3,230,2,96,198,2,96
310 DATA-1

READY.
```

Raster demonstration prog. 1 (splitting screen colours)
Assembly language listing

```
.: C000 20 09 C0
A C000 JSR $C009
: C003 20 6F C0
A C003 JSR $C06F
: C006 4C 03 C0
A C006 JMP $C003
: C009 78
A C009 SEI
: C00A AD 0E DC
A C00A LDA $DC0E
: C00D 29 FE
A C00D AND #$FE
: C00F 8D 0E DC
A C00F STA $DC0E
```

STOP TIMER *A* ON C.I.A: # 1 which will in turn disable the 1/60th second HARDWARE IRQ of the CBM 64.

```
: C012 AD 11 D0
A C012 LDA $D011
: C015 29 7F
A C015 AND #$7F
: C017 8D 11 D0
A C017 STA $D011
```

OMIT High bit from Raster COUNT.

```
: C01A A9 90
A C01A LDA #$90
: C01C 8D 12 D0
A C01C STA $D012
```

Set Raster compare Register for approximately halfway down the screen.

```
: C01F 85 02
A C01F STA $02
```

Save this value in zero page for later use.

```
: C021 A9 40
A C021 LDA #$40
: C023 8D 14 03
A C023 STA $0314
: C026 A9 C0
A C026 LDA #$C0
: C028 8D 15 03
A C028 STA $0315
```

Point the CBM 64's HARDWARE IRQ vector towards the Raster Routine ie. $C040

```
: C02B A9 93
A C02B LDA #$93
: C02D 20 D2 FF
A C02D JSR $FFD2
```

Clear screen.

```
: C030 AD 1A D0
A C030 LDA $D01A
: C033 09 01
A C033 ORA #$01
: C035 8D 1A D0
A C035 STA $D01A
```

Enable bit 0 of the Raster Interrupt enable register which will cause an interrupt from the raster register to be recognised

```
: C038 58
A C038 CLI
: C039 60
A C039 RTS
: C03A 68
A C03A PLA
: C03B AA
A C03B TAX
: C03C 68
A C03C PLA
: C03D A8
A C03D TAY
: C03E 68
A C03E PLA
: C03F 40
A C03F RTI
```

Pull all of the three registers (A,X & Y) off the stack in the correct order ie. FIRST on/last off and vice versa.

Then return from an interrupt.

```
: C040 A9 01
A C040 LDA #$01
: C042 2C 19 D0
A C042 BIT $D019
```

Check if bit 0 of the interrupt status register is set.

Program 1 Listing (cont.)

```
A C042 BIT $D019
: C045 F0 F3
A C045 BEQ $C03A
```
If it is not, branch back to exit the interrupt.
```
: C047 AD 21 D0
A C047 LDA $D021
: C04A 29 0F
A C04A AND #$0F
```
Get current screen colour and exclude the high nibble (4 bits), therefore ensuring that the colour = (0 - 15).
```
: C04C C9 06
A C04C CMP #$06
```
Is the screen coloured Blue?
```
: C04E D0 15
A C04E BNE $C065
```
No it is not, so branch to $C065
```
: C050 A9 01
A C050 LDA #$01
: C052 8D 21 D0
A C052 STA $D021
```
Yes it is, so make it white instead.
```
: C055 A9 00
A C055 LDA #$00
: C057 8D 12 D0
A C057 STA $D012
```
Reset the Raster compare register to the top of the screen.
```
: C05A AD 19 D0
A C05A LDA $D019
: C05D 09 01
A C05D ORA #$01
: C05F 8D 19 D0
A C05F STA $D019
```
Signal to the raster register that the interrupt is now complete.
```
: C062 4C 3A C0
A C062 JMP $C03A
```
Jump to $C03A : ie. pull the registers and return.
```
: C065 A9 06
A C065 LDA #$06
: C067 8D 21 D0
A C067 STA $D021
```
Change screen value to blue.
```
: C06A A5 02
A C06A LDA $02
: C06C 4C 57 C0
A C06C JMP $C057
```
Load accumulator with secondary raster position and jump to $C057.
```
: C06F 20 9F FF
A C06F JSR $FF9F
```
JSR $ SCNKey ie. Read keyboard. This is necessary because the normal hardware IRQ which usually scans the keyboard has been blocked.
```
: C072 A5 C5
A C072 LDA $C5
```
Get value of key pressed.
```
: C074 C9 07
A C074 CMP #$07
```
Is it cursor up/down?
```
: C076 F0 01
A C076 BEQ $C079
```
If so, then branch to $C079
```
: C078 60
A C078 RTS
```
Else return from subroutine. Rem: subroutine begins at $C06F.
```
: C079 AD 8D 02
A C079 LDA $028D
: C07C 29 01
A C07C AND #$01
```
Is the 'SHIFT' key flag set?
```
: C07E D0 03
A C07E BNE $C083
```
Yes, then branch to $C083
```
: C080 E6 02
A C080 INC $02
```
No, increment the secondary raster position (ie move split down).
```
: C082 60
A C082 RTS
```
Return from subroutine.
```
: C083 C6 02
A C083 DEC $02
```
Decrement secondary raster position (ie. move split up).
```
: C085 60
A C085 RTS
```
Return from sub-routine.
```
: C086 00

READY.
```

Program 2 Listing

```
10 REM ***********************************
20 REM * RASTER DEMONSTRATION PROG#2 *
30 REM *                                 *
40 REM * MIXING USER DEFINED GRAPHICS*
50 REM * WITH STANDARD PET CHARACTERS*
60 REM ***********************************
70 FORT=0TO7:READA:POKE(4096*3)+T,A:NEXTT:REM SET UP ONE USER GRAPHIC FOR DEMO
80 DATA0,120,255,194,192,194,255,120:REM GRAPHIC DATA
90 FORT=0TO7:POKE(4096*3)+(32*8)+T,0:NEXT
100 TL=0:C=0:SA=49152
110 READA:IFA=-1THEN140
120 POKESA+C,A:C=C+1:TL=TL+A
130 GOTO110
140 PRINT"♥";:REM * SHIFT & CLR/HOME *
150 IFC=139THEN190
160 IFC<139THENPRINT"TOO FEW ";:GOTO180
170 PRINT"TOO MANY ";
180 PRINT"DATA ITEMS▒":REM * CU DOWN *
190 IFTL=15856THEN220
200 PRINT"ERROR IN DATA INPUT"
210 STOP
220 PRINT"♥PRESS ANY KEY TO SEE DEMONSTRATION."
230 PRINT"▒▒PRESS 'CURSOR UP/DOWN' WITH OR WITHOUT  ":REM * CU DOWN X 2 *
240 PRINT"'SHIFT', TO MOVE THE SPLIT UP OR DOWN    "
250 PRINT"THE SCREEN."
260 POKE198,0:WAIT198,1
270 PRINT"♥";:FORT=0TO23:PRINT"@ @ @ @ @ @ @ @ @ @ @ @ @ @ @ @ @ @ @ @ ";:NEXT
280 SYS49152
300 DATA32,9,192,32,116,192,76,3,192,120,173,14,220,41,254,141,14,220,173,17
310 DATA208,41,127,141,17,208,169,144,141,18,208,133,2,169,69,141,20,3,169,192
320 DATA141,21,3,234,234,234,234,234,173,26,208,9,1,141,26,208,169,20,141,24
330 DATA208,88,96,104,170,104,168,104,64,169,1,44,25,208,240,243,173,24,208,41
340 DATA30,201,20,208,21,169,28,141,24,208,169,0,141,18,208,173,25,208,9,1,141
350 DATA25,208,76,63,192,169,20,141,24,208,165,2,76,92,192,32,159,255,165,197
360 DATA201,7,240,1,96,173,141,2,41,1,208,3,230,2,96,198,2,96
370 DATA-1
```

Raster demonstration prog. 2 (Assembler listing)
Mixing user defined graphics with PET characters

```
.: C000 20 09 C0
A C000 JSR $C009
: C003 20 74 C0
A C003 JSR $C074
: C006 4C 03 C0
A C006 JMP $C003
: C009 78
A C009 SEI
: C00A AD 0E DC
A C00A LDA $DC0E
: C00D 29 FE
A C00D AND #$FE
: C00F 8D 0E DC
A C00F STA $DC0E
```
Stop timer a of C.I.A. # 2 which will in turn disable the 1/60th second hardware IRQ of the CBM 64.
```
: C012 AD 11 D0
A C012 LDA $D011
: C015 29 7F
```
Omit the high bit from the raster count.

Included with each of the four demonstrations is a fully documented Assembly language listing. These should be studied carefully until such time as you get an understanding of the procedure involved; by then you will be ready to experiment on your own.

Raster interrupt registers

There are three very important registers used with raster interrupts: the raster compare register ($D012), the interrupt status register ($D019) and the interrupt enable register.

The raster itself scans the screen and is visible between the dot positions of 50 and 250, which is 200 dot positions altogether or 25 lines x 8 dots.

To obtain a flicker free screen display, the raster compare register should be

Program 2 Listing (cont.)

```
A C015 AND #$7F
: C017 8D 11 D0
A C017 STA $D011
: C01A A9 90
A C01A LDA #$90
: C01C 8D 12 D0
A C01C STA $D012
```

Set the raster compare for approximately halfway down the screen.

```
: C01F 85 02
A C01F STA $02
```

Save value in zero page for later use.

```
: C021 A9 45
A C021 LDA #$45
: C023 8D 14 03
A C023 STA $0314
: C026 A9 C0
A C026 LDA #$C0
: C028 8D 15 03
A C028 STA $0315
```

Point the CBM 64's hardware IRQ vector to point towards the raster routine ie. $C045

```
: C02B EA
A C02B NOP
: C02C EA
A C02C NOP
: C02D EA
A C02D NOP
: C02E EA
A C02E NOP
: C02F EA
A C02F NOP
: C030 AD 1A D0
A C030 LDA $D01A
: C033 09 01
A C033 ORA #$01
: C035 8D 1A D0
A C035 STA $D01A
```

Set bit 0 of the interrupt enable register therefore enabling an interrupt from the raster register, to be recognised.

```
: C038 A9 14
A C038 LDA #$14
: C03A 8D 18 D0
A C03A STA $D018
```

Ensure that present characters being displayed are PET characters.

```
: C03D 58
A C03D CLI
: C03E 60
A C03E RTS
: C03F 68
A C03F PLA
: C040 AA
A C040 TAX
: C041 68
A C041 PLA
: C042 A8
A C042 TAY
: C043 68
A C043 PLA
: C044 40
A C044 RTI
```

Pull the three register values (A, X & Y) off the stack in the correct order, ie. first on last off and vice versa then return from an interrupt.

```
: C045 A9 01
A C045 LDA #$01
: C047 2C 19 D0
A C047 BIT $D019
```

Check if current interrupt being processed was caused by the raster register

```
: C04A F0 F3
A C04A BEQ $C03F
```

If it was not, branch to $C03F ie. exit from interrupt

```
: C04C AD 18 D0
A C04C LDA $D018
: C04F 29 1E
A C04F AND #$1E
```

Get status of current characters being displayed.

```
: C051 C9 14
A C051 CMP #$14
```

Are they CBM PET characters?

```
: C053 D0 15
A C053 BNE $C06A
```

No, so branch to $C06A

```
: C055 A9 1C
A C055 LDA #$1C
: C057 8D 18 D0
A C057 STA $D018
```

Yes they are, so switch to user defined graphics.

```
: C05A A9 00
A C05A LDA #$00
: C05C 8D 12 D0
A C05C STA $D012
```

Reset raster compare register to the top of the screen.

```
: C05F AD 19 D0
A C05F LDA $D019
: C062 09 01
A C062 ORA #$01
: C064 8D 19 D0
A C064 STA $D019
```

Signal the interrupt status register, that the interrupt has been carried out.

```
: C067 4C 3F C0
A C067 JMP $C03F
```

Jump to $C03F ie. pull the registers off the stack and return

```
: C06A A9 14
A C06A LDA #$14
: C06C 8D 18 D0
A C06C STA $D018
```

Change current character display status to standard CBM PET characters

```
: C06F A5 02
A C06F LDA $02
: C071 4C 5C C0
A C071 JMP $C05C
```

Load accumulator with the secondary raster position and jump to $C05C.

```
: C074 20 9F FF
A C074 JSR $FF9F
```

JSR $ SCN Key ie. read keyboard. This routine is necessary because the normal hardware IRQ routine which scans the keyboard has been blocked.

```
: C077 A5 C5
A C077 LDA $C5
```

Get value of current key pressed.

```
: C079 C9 07
A C079 CMP #$07
```

Is it cursor up/down?

```
: C07B F0 01
A C07B BEQ $C07E
```

If so, then branch to $C07E

```
: C07D 60
A C07D RTS
```

Else return from subroutine : rem subroutine beins at $C074.

```
: C07E AD 8D 02
A C07E LDA $028D
: C081 29 01
A C081 AND #$01
```

Is the shift key flag set?

```
: C083 D0 03
A C083 BNE $C088
```

If so, then branch to $C088

```
: C085 E6 02
A C085 INC $02
: C087 60
A C087 RTS
```

Else increment the secondary raster position; ie move the split down the screen & return from subroutine.

```
: C088 C6 02
A C088 DEC $02
: C08A 60
A C08A RTS
```

Decrement the secondary raster position; ie. move the split up the screen & return from subroutine.

Program 3

```
10 REM ********************************
20 REM * RASTER DEMONSTRATION PROG#3 *
30 REM *                              *
40 REM * MIXING TEXT & HI RESOLUTION *
50 REM ********************************
60 TL=0:C=0:SA=49152
70 READA:IFA=-1THEN100
80 POKESA+C,A:C=C+1:TL=TL+A
90 GOTO70
100 PRINT"[CLR]";:REM * SHIFT & CLR/HOME *
110 IFC=149THEN150
120 IFC<149THENPRINT"TOO FEW ";:GOTO140
130 PRINT"TOO MANY ";
140 PRINT"DATA ITEMS[CD]":REM * CU DOWN *
150 IFTL=16443THEN180
160 PRINT"ERROR IN DATA INPUT"
170 STOP
180 PRINT"[CLR]PRESS ANY KEY TO SEE DEMONSTRATION."
190 PRINT"[CD][CD]PRESS 'CURSOR UP/DOWN' WITH OR WITHOUT  "
200 PRINT"'SHIFT', TO MOVE THE SPLIT UP OR DOWN    "
210 PRINT"THE SCREEN."
```

set to a value which is out of the 'visible patch', eg, 0-49 (incl) and 250-255 (incl).

The interrupt status register, when read, gives the current status of interrupts, ie, if any bit set in the interrupt status register corresponds with a bit set in the interrupt enable register, then an interrupt from that source will take or is already taking, place.

The appropriate bits of the interrupt status register are as follows:

Bit	Condition
0	Set when raster compare reaches the preset value.
1	Set when sprite collides with background
2	Set when sprite collides with another sprite
3	Triggered by light pen/rifle.
4	This bit is set whenever any of the above bits are set.

It is interesting to note that it is standard procedure for the 'VIC chip' to execute

interrupts of bits 1&2, ie, sprite to sprite data collisions and sprite to sprite collisions.

Finally, the interrupt enable register ($D01A) which receives the interrupt request should be used carefully in order to achieve the desired results. Examine the bits above, pick the type of interrupt you wish to call, and set the appropriate bit in the interrupt enable register to allow an interrupt from that source to occur.

Program 3 Listing (cont.)

```
220 POKE198,0:WAIT198,1
230 PRINT"J";:REM * SHIFT & CLR/HOME *
240 PRINT"THE FIRST COUPLE OF LINES OF THE SCREEN CONTAIN NOTHING BUT TEXT......
.";
250 PRINT"........USE THE CURSOR TO SEE THE TEXT  BEING   COVERED BY THE HI-";
260 PRINT"RESOLUTION SCREEN."
270 SYS49152
300 DATA32,9,192,32,126,192,76,3,192,120,173,14,220,41,254,141,14,220,173,17
310 DATA208,41,127,141,17,208,169,144,141,18,208,133,2,169,59,141,20,3,169,192
320 DATA141,21,3,173,26,208,9,1,141,26,208,88,96,104,170,104,168,104,64,169,1
330 DATA44,25,208,240,243,173,24,208,41,8,208,32,173,17,208,9,32,141,17,208,173
340 DATA24,208,9,8,141,24,208,169,0,141,18,208,173,25,208,9,1,141,25,208,76,53
350 DATA192,173,17,208,41,223,141,17,208,173,24,208,41,247,141,24,208,165,2,76
360 DATA91,192,32,159,255,165,197,201,7,240,1,96,173,141,2,41,1,208,3,230,2,96
370 DATA198,2,96
380 DATA-1
```

Raster demonstration prog 3 (mixing text & Hi res)
Assembly language listing

```
.: C000 20 09 C0
A C000 JSR $C009
: C003 20 7E C0
A C003 JSR $C07E
: C006 4C 03 C0
A C006 JMP $C003
: C009 78
A C009 SEI
: C00A AD 0E DC
A C00A LDA $DC0E
: C00D 29 FE
A C00D AND #$FE
: C00F 8D 0E DC
A C00F STA $DC0E
```

Stop time A on C.I.A. # 1 which will in turn disable the 1/60th second hardware IRQ of the CBM 64.

```
: C012 AD 11 D0
A C012 LDA $D011
: C015 29 7F
A C015 AND #$7F
: C017 8D 11 D0
A C017 STA $D011
```

Omit the high bit from the raster count.

```
: C01A A9 90
A C01A LDA #$90
: C01C 8D 12 D0
A C01C STA $D012
```

Set the raster compare register for approximately halfway down the screen.

```
: C01F 85 02
A C01F STA $02
```

Save this value in zero page for later use.

```
: C021 A9 3B
A C021 LDA #$3B
: C023 8D 14 03
A C023 STA $0314
: C026 A9 C0
A C026 LDA #$C0
: C028 8D 15 03
A C028 STA $0315
```

Point the CBM 64's hardware IRQ vector towards the raster routine, ie. $C 03B

```
: C02B AD 1A D0
A C02B LDA $D01A
: C02E 09 01
A C02E ORA #$01
: C030 8D 1A D0
A C030 STA $D01A
```

Set bit 0 of the interrupt enable register to allow raster interrupts to take place.

```
: C033 58
A C033 CLI
: C034 60
A C034 RTS
: C035 68
A C035 PLA
: C036 AA
A C036 TAX
: C037 68
A C037 PLA
: C038 A8
A C038 TAY
: C039 68
A C039 PLA
: C03A 40
A C03A RTI
```

Pull the three registers; A,X & Y off the stack in the correct order ie. first on, last off and vice versa, then return from an interrupt.

```
: C03B A9 01
A C03B LDA #$01
: C03D 2C 19 D0
A C03D BIT $D019
```

Check if bit 0 of the interrupt status is set. If it is then a raster interrupt has occurred

```
: C040 F0 F3
A C040 BEQ $C035
```

If it is not then branch to $ CO35, ie. pull registers & return.

```
: C042 AD 18 D0
A C042 LDA $D018
: C045 29 08
A C045 AND #$08
```

Check if hi-resolution screen is already present.

```
: C047 D0 20
A C047 BNE $C069
```

If it is, then branch to $C069.

```
: C049 AD 11 D0
A C049 LDA $D011
: C04C 09 20
A C04C ORA #$20
: C04E 8D 11 D0
A C04E STA $D011
```

Enable bit map mode

```
: C051 AD 18 D0
A C051 LDA $D018
: C054 09 08
A C054 ORA #$08
: C056 8D 18 D0
A C056 STA $D018
```

Shift high resolution screen to 8K + ie. 8192 + or $2000

```
: C059 A9 00
A C059 LDA #$00
: C05B 8D 12 D0
A C05B STA $D012
```

Reset raster compare to top of screen.

```
: C05E AD 19 D0
A C05E LDA $D019
: C061 09 01
A C061 ORA #$01
: C063 8D 19 D0
A C063 STA $D019
```

Signal to the raster register that the interrupt has been executed.

```
: C066 4C 35 C0
A C066 JMP $C035
```

Jump to $C035 ie. pull registers off stack and return

```
: C069 AD 11 D0
A C069 LDA $D011
: C06C 29 DF
A C06C AND #$DF
: C06E 8D 11 D0
A C06E STA $D011
```

Disable bit map mode.

```
: C071 AD 18 D0
A C071 LDA $D018
: C074 29 F7
A C074 AND #$F7
: C076 8D 18 D0
A C076 STA $D018
```

Reset character dot address to default value.

```
: C079 A5 02
A C079 LDA $02
: C07B 4C 5B C0
A C07B JMP $C05B
```

Load accumulator with secondary raster position and jump to £C05B

```
: C07E 20 9F FF
A C07E JSR $FF9F
```

JSR $ SCN Key ie. read keyboard. This routine must be called because the normal hardware IRQ routine which scans the keyboard has been blocked.

```
: C081 A5 C5
A C081 LDA $C5
```

Get value of last key pressed.

```
: C083 C9 07
A C083 CMP #$07
```

Is it cursor up/down?

```
: C085 F0 01
A C085 BEQ $C088
```

If it is then branch to $C088

```
: C087 60
A C087 RTS
```

Else return from subroutine. rem: subroutine starts at $C07E

```
: C088 AD 8D 02
A C088 LDA $028D
: C08B 29 01
A C08B AND #$01
```

Check if 'shift' key flag is set

```
: C08D D0 03
A C08D BNE $C092
```

If it is then branch to $C092

```
: C08F E6 02
A C08F INC $02
: C091 60
A C091 RTS
```

Otherwise increment the secondary raster position ie. move split down the screen and return from the subroutine.

```
: C092 C6 02
A C092 DEC $02
: C094 60
A C094 RTS
```

Decrement the secondary raster position, ie. move the split up the screen & return from the subroutine.

Program 4 Listing

```
10 REM ********************************
20 REM * RASTER DEMONSTRATION PROG#4 *
30 REM *                              *
40 REM * DISPLAYING MORE THAN EIGHT   *
50 REM * M.O.B'S (SPRITES) ON SCREEN *
60 REM ********************************
70 TL=0:C=0:SA=49152
80 READA:IFA=-1THEN110
90 POKESA+C,A:C=C+1:TL=TL+A
100 GOTO80
110 PRINT"♥";:REM * SHIFT & CLR/HOME *
120 IFC=227THEN160
130 IFC<227THENPRINT"TOO FEW ";:GOTO150
140 PRINT"TOO MANY ";
150 PRINT"DATA ITEMS▮":REM * CU DOWN *
160 IFTL=28602THEN190
170 PRINT"ERROR IN DATA INPUT"
180 STOP
190 PRINT"♥PRESS ANY KEY TO SEE DEMONSTRATION."
200 PRINT"▮▮PRESS 'CURSOR UP/DOWN' WITH OR WITHOUT  "
210 PRINT"'SHIFT', TO MOVE THE 64 SPRITES UP OR     "
220 PRINT"DOWN THE SCREEN."
230 POKE198,0:WAIT198,1
240 SYS49152
300 DATA32,9,192,32,200,192,76,3,192,120,173,14,220,41,254,141,14,220,173,17,208
310 DATA41,127,141,17,208,162,64,169,255,157,63,3,202,208,250,162,0,169,24,157
320 DATA0,208,24,105,32,232,232,224,16,208,244,162,8,169,13,157,247,7,169,1,157
330 DATA38,208,202,208,243,169,255,141,21,208,162,0,169,50,157,1,208,232,232,224
340 DATA16,208,247,169,0,133,253,169,50,141,18,208,133,254,169,122,141,20,3,169
350 DATA192,141,21,3,173,26,208,9,1,141,26,208,88,96,104,170,104,168,104,64,169
360 DATA1,44,25,208,240,243,234,165,254,24,105,2,141,1,208,141,3,208,141,5,208
370 DATA141,7,208,141,9,208,141,11,208,141,13,208,141,15,208,166,253,232,224,8
380 DATA240,23,134,253,165,254,24,105,24,133,254,141,18,208,173,25,208,9,1,141
390 DATA25,208,76,116,192,169,0,133,253,165,2,133,254,76,175,192,32,159,255,120
400 DATA165,197,201,7,240,2,88,96,173,141,2,41,1,208,4,230,2,88,96,198,2,88,96
410 DATA-1
```

Raster demonstration Prog 4 (displaying more than 8 sprites) Assembly language listing.

```
: C000 20 09 C0
A C000 JSR $C009
: C003 20 C8 C0
A C003 JSR $C0C8
: C006 4C 03 C0
A C006 JMP $C003
: C009 78
A C009 SEI
: C00A AD 0E DC
A C00A LDA $DC0E
: C00D 29 FE
A C00D AND #$FE
: C00F 8D 0E DC
A C00F STA $DC0E
```

Stop timer A on C.I.A. 1 which will in turn disable the 1/60th second hardware IRQ of the CBM 64.

```
: C012 AD 11 D0
A C012 LDA $D011
: C015 29 7F
A C015 AND #$7F
: C017 8D 11 D0
A C017 STA $D011
```

Omit the high bit from the raster count.

```
: C01A A2 40
A C01A LDX #$40
: C01C A9 FF
A C01C LDA #$FF
: C01E 9D 3F 03
A C01E STA $033F,X
: C021 CA
A C021 DEX
: C022 D0 FA
A C022 BNE $C01E
```

Set up a box shaped sprite in block no. 13 eg. 13 x 64 = 832 + for display purposes

```
: C024 A2 00
A C024 LDX #$00
: C026 A9 18
A C026 LDA #$18
: C028 9D 00 D0
A C028 STA $D000,X
: C02B 18
A C02B CLC
: C02C 69 20
A C02C ADC #$20
: C02E E8
A C02E INX
: C02F E8
A C02F INX
: C030 E0 10
A C030 CPX #$10
: C032 D0 F4
A C032 BNE $C028
```

Set up the x co-ordinates of each of the eight sprites, starting at bit position 24 (decimal) and progressing in steps of 32 (decimal)

eg. SPQ	SP1X	SP2X	SP3X	etc
24	56	88	120	

```
: C034 A2 08
A C034 LDX #$08
: C036 A9 0D
A C036 LDA #$0D
: C038 9D F7 07
A C038 STA $07F7,X
: C03B A9 01
A C03B LDA #$01
: C03D 9D 26 D0
A C03D STA $D026,X
: C040 CA
A C040 DEX
: C041 D0 F3
A C041 BNE $C036
```

Set the sprite pointers of all the eight sprites to block no. 13

Also make all the sprite colours white.

```
: C043 A9 FF
A C043 LDA #$FF
: C045 8D 15 D0
A C045 STA $D015
```

Enable all of the eight sprites

```
: C048 A2 00
A C048 LDX #$00
: C04A A9 32
A C04A LDA #$32
: C04C 9D 01 D0
A C04C STA $D001,X
: C04F E8
A C04F INX
: C050 E8
A C050 INX
: C051 E0 10
A C051 CPX #$10
: C053 D0 F7
A C053 BNE $C04C
```

Set Y co-ordinates of all the eight sprites to 50 decimal

```
: C055 A9 00
A C055 LDA #$00
: C057 85 FD
A C057 STA $FD
```

Set sprite count flag to zero

```
: C059 A9 32
A C059 LDA #$32
: C05B 8D 12 D0
A C05B STA $D012
: C05E 85 FE
A C05E STA $FE
```

Set raster compare value to 50 decimal and save value in zero page for later use.

Final Note

Once an interrupt has been carried out, it is important to write a '1' back to the correct bit in the interrupt status register ($D019). This signifies that the interrupt has been processed.

The format for each interrupt is basically the same; read the Assembly listing carefully and you should get to understand the method used in setting up raster interrupts. Do not be afraid to experiment. Start with simple routines and gradually work your way to using multiple interrupts. Remember you can but try.

Conversion hints

As all of the four routines use the Commodore 64's extensive graphics facilities, it is not even worth thinking about converting to any other micro, unless VIC 20 users think they can make the necessary changes in each routine, excluding number 4, as the VIC 20 does not support 'Sprite Graphics'.

Program 4 Listing

```
: C060 A9 7A
A C060 LDA #$7A
: C062 8D 14 03
A C062 STA $0314
: C065 A9 C0
A C065 LDA #$C0
: C067 8D 15 03
A C067 STA $0315
```

Point the CBM 64's hardware vector towards the raster routine ie. $CE7A.

```
: C06A AD 1A D0
A C06A LDA $D01A
: C06D 09 01
A C06D ORA #$01
: C06F 8D 1A D0
A C06F STA $D01A
```

Set bit 0 of the interrupt enable register to allow raster interrupts to take place.

```
: C072 58
A C072 CLI
: C073 60
A C073 RTS
: C074 68
A C074 PLA
: C075 AA
A C075 TAX
: C076 68
A C076 PLA
: C077 A8
A C077 TAY
: C078 68
A C078 PLA
: C079 40
A C079 RTI
```

Pull the three registers: A,X,& Y off the stack in the correct order ie. first on/last off and vice versa, then return from an interrupt.

```
: C07A A9 01
A C07A LDA #$01
: C07C 2C 19 D0
A C07C BIT $D019
```

Check if bit 0 of the interrupt status register is set, if so then a raster interrupt has occurred.

```
: C07F F0 F3
A C07F BEQ $C074
```

If not, then branch back to $C074 ie. exit interrupt

```
: C081 EA
A C081 NOP
: C082 A5 FE
A C082 LDA $FE
: C084 18
A C084 CLC
: C085 69 02
A C085 ADC #$02
: C087 8D 01 D0
A C087 STA $D001
: C08A 8D 03 D0
A C08A STA $D003
: C08D 8D 05 D0
A C08D STA $D005
: C090 8D 07 D0
A C090 STA $D007
: C093 8D 09 D0
A C093 STA $D009
: C096 8D 0B D0
A C096 STA $D00B
: C099 8D 0D D0
A C099 STA $D00D
: C09C 8D 0F D0
A C09C STA $D00F
```

Get current raster position, add two to avoid flicker and store value in the Y co-ordinates of all the eight sprites.

```
: C09F A6 FD
A C09F LDX $FD
: C0A1 E8
A C0A1 INX
: C0A2 E0 08
A C0A2 CPX #$08
```

Check if sprite count flag has reached 8!

```
: C0A4 F0 17
A C0A4 BEQ $C0BD
```

If so then all eight sprites have been displayed 8 times on screen so branch to $C0BD

```
: C0A6 86 FD
A C0A6 STX $FD
```

If not then store value back in zero page.

```
: C0A8 A5 FE
A C0A8 LDA $FE
: C0AA 18
A C0AA CLC
: C0AB 69 18
A C0AB ADC #$18
: C0AD 85 FE
A C0AD STA $FE
: C0AF 8D 12 D0
A C0AF STA $D012
```

Add 24 decimal to the raster compare value to get the next sprite display co-ordinate and store it in zero page and the raster compare register.

```
: C0B2 AD 19 D0
A C0B2 LDA $D019
: C0B5 09 01
A C0B5 ORA #$01
: C0B7 8D 19 D0
A C0B7 STA $D019
```

Set bit 0 of the interrupt status register to signal that the raster interrupt has been completed.

```
: C0BA 4C 74 C0
A C0BA JMP $C074
```

Jump to $C074 ie. Exit interrupt.

```
: C0BD A9 00
A C0BD LDA #$00
: C0BF 85 FD
A C0BF STA $FD
: C0C1 A5 02
A C0C1 LDA $02
: C0C3 85 FE
A C0C3 STA $FE
: C0C5 4C AF C0
A C0C5 JMP $C0AF
```

Sprites have been displayed 8 times so reset the sprite count flag to zero and reset the raster compare value.

```
: C0C8 20 9F FF
A C0C8 JSR $FF9F
```

JSR $ SCN Key. This routine is necessary because the normal key scan routine has been blocked.

```
: C0CB 78
A C0CB SEI
: C0CC A5 C5
A C0CC LDA $C5
```

Get value of last key pressed

```
: C0CE C9 07
A C0CE CMP #$07
```

Is it cursor up/down?

```
: C0D0 F0 02
A C0D0 BEQ $C0D4
```

If yes, then branch to $C0D4

```
: C0D2 58
A C0D2 CLI
: C0D3 60
A C0D3 RTS
```

Else return from subroutine. Rem: subroutine starts at $ COC8.

```
: C0D4 AD 8D 02
A C0D4 LDA $028D
: C0D7 29 01
A C0D7 AND #$01
```

Is 'shift' key flag set?

```
: C0D9 D0 04
A C0D9 BNE $C0DF
```

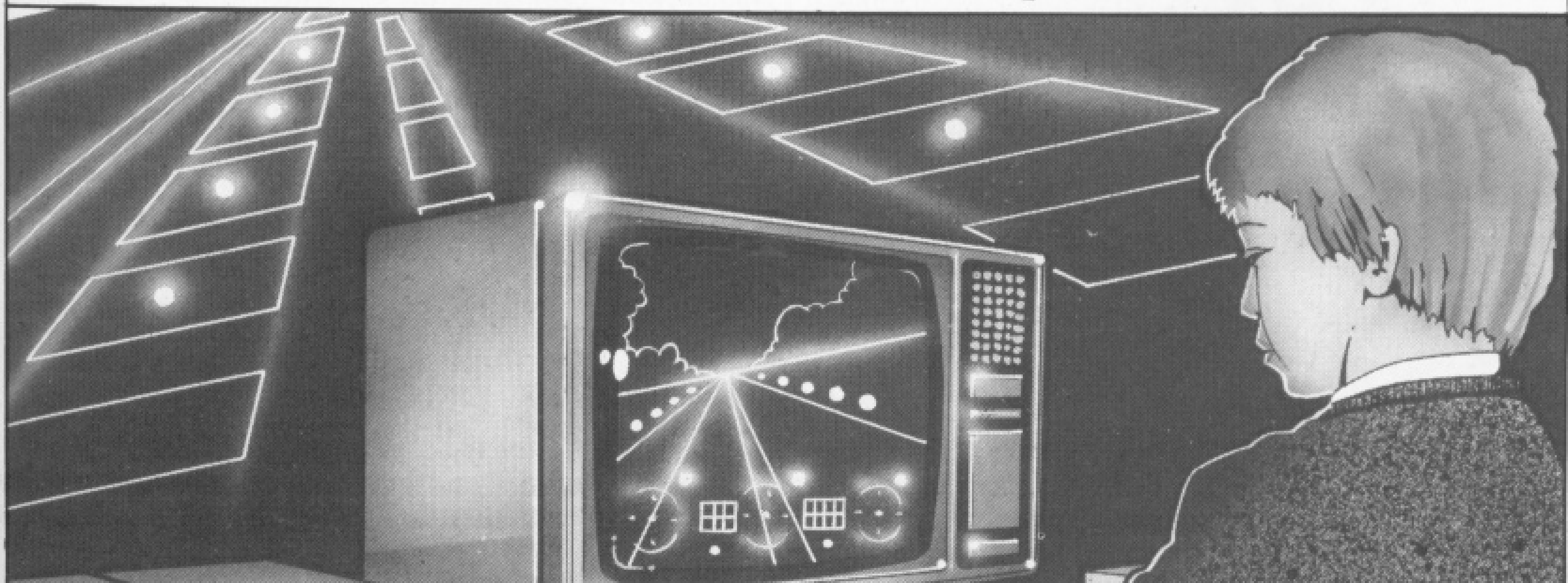
If yes, then branch to $C0DF

```
: C0DB E6 02
A C0DB INC $02
: C0DD 58
A C0DD CLI
: C0DE 60
A C0DE RTS
```

Else return from subroutine. Rem: subroutine starts at $C0C8.

```
: C0DF C6 02
A C0DF DEC $02
: C0E1 58
A C0E1 CLI
: C0E2 60
A C0E2 RTS
```

Decrement raster value ie. move sprites up screen then return from subroutine.



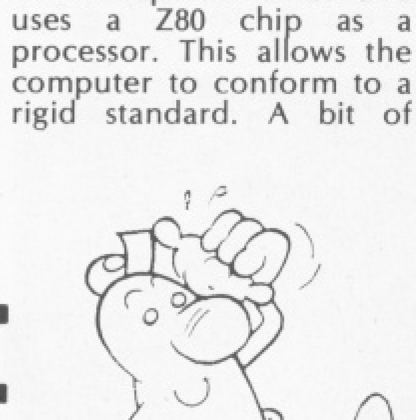
INPUT

Resident 'agony uncle', Simon Rockman endeavours to solve your Commodore problems.

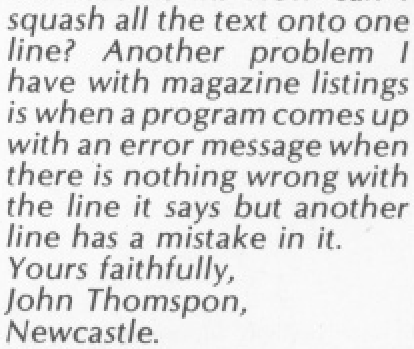


Dear Sir,
What is CP/M? If I buy the Commodore 64 CP/M cartridge can I use software from other machines?
Yours faithfully,
A Campbell,
Glasgow.

We answer,
CP/M is an operating system. The ROM of a 64 can be roughly divided up into two sections, Basic and Operating Systems; in the 64 it is known as the kernal. CP/M replaces these and uses a Z80 chip as a processor. This allows the computer to conform to a rigid standard. A bit of clever software, called the NIOS, tells CP/M how your machine is configured, that it has a screen, disc and a keyboard, and translates from the 64 hardware to the CP/M software. Every machine that runs CP/M has its own BIOS which links into standard CP/M. To CP/M all computers look the same. This means that any program written to conform to the CP/M standard will work on any of the many CP/M computers. There is one major drawback. This is the lack of a standard for a disc format. Because the hardware for disc drives has improved so rapidly it is not possibly to read any disc on any drive. This is especially true of the 1541 which is totally non-standard. To run any CP/M software using the '64 cartridge it is necessary to find a supplier with the correct disc format. These are very few and far between. The other major problem with CP/M discs is that a lot of CP/M programs expect dual disc drives. The Commodore system cannot cater for this. CP/M usually requires an 80 column display, this means that any program which uses this cannot be run on a 64. The most useful application left for CP/M on the '64 is for running languages, CP/M provides for all the major languages; all you have to worry about is disc size and format. Still at £50 the Commodore cartridge is very cheap.



Dear Sir,
When typing in programs from magazines I often find that the lines in the listing will not fit in. How can I squash all the text onto one line? Another problem I have with magazine listings is when a program comes up with an error message when there is nothing wrong with the line it says but another line has a mistake in it.
Yours faithfully,
John Thomspon,
Newcastle.



We answer,
Since the early days of Commodore computers life has been made easy for the programmer by the use of shorthand commands. The most common of these is the use of ? for PRINT, but there are a whole bunch of these commands which are listed in the back of your manual (page 130 if you have a Commodore 64). The Commodore screen editor will allow you to put eighty characters on each line (88 on a VIC) but the use of abbreviations will cause the line to expand past this. Try this line:-

```
1?:?:?:?:?:?:?:?:?:?:?:?:?:?:
?:?:?:?:?:?:?:?:?:?:?:?:?:?:?:
?:?:?:?:?:?:?:?:?:?:?
```

you will need to move the cursor back to hit return over it. When you list the line it will have expanded out to six lines (more on a VIC). This is useful for cramming subroutines onto one line but makes the program very hard to work on because each time you want to change the line you have to convert each keyword back to its shorthand form (or token). For this reason the practice should be avoided, if necessary split the line up into two short lines. This will be slower but easier to work with.

The problem of the computer reporting the error message is one which confuses a lot of first time users. The computer complains when it detects the error not necessarily when the error occurs. The usual problem with magazine programs occurs with the READ and DATA statements. It is quite common to have a short loop within a program like this:-

```
510 FOR JJ=Z TO Z+8:READ
NU:POKE JJ,NU :NEXT
520 DATA 12,42,95,11,49,1,
52,252
```

This routine is perfectly OK (it won't work here because it is only part of a program so don't type it in). If you should miss a comma from line 520 it will show up as an error in line 510. This will take the form of either an ?illegal quantity error or an ?out of data error. The former will happen if missing a comma has caused the program to READ a number greater that 255 (say the comma between the 11 and the 49), and the later

error will occur if all the other values are OK but it runs out of data. The ?illegal quantity error is produced by the POKE trying to use too large a number. The computer cannot guess what you are trying to read the data for and so will never produce an error in a data line, except when the syntax is wrong. So be careful when using the line number from an error in ...report.



Dear Sir,
I want to put my name in a box at the start of a program but whenever I use a REM and the graphics symbols I get a load of keywords that I don't want, how can I fix this?
Yours faithfully,
Simon Jenson,
London

We answer,
In another letter we talked about tokens, the short form of keywords; you have come across a bug in Commodore BASIC. It takes any shifted character and converts it into a keyword. This is particularly noticeable in lower case mode. Press the Commodore key and shift to go into lower case and then type:-

1 rem Your Commodore.

When you list it you get:-

1 rem gosubour lenomore



The computer has converted the Y into a gosub and the C into a len. There is a way around this, use quotes:-
1 rem "Your Commodore will always list correctly, so to draw a box around your name start each line with a 'REM" '; there is no need to close the quotes.

Dear Sir,
I am having an argument with a friend, he says that you should always have a letter after a NEXT, but I know that it works without, who is right?
Yours faithfully,
J Rawlings,
Bath.

We answer,
Both of you! If you miss out the variable name after a NEXT in a FOR...NEXT loop the computer will look back to see if it is in the middle of any loops and assuming it is it will use that variable, so you are right. However don't get big headed, there is a school of thought, known as structured programming which says that not only must programs work but they must be easy to work on. That means everything should be laid out clearly. If you have a lot of FOR...NEXT loops it can get confusing as to which FOR a NEXT is referring to. For this reason it is good practice to label the NEXT. There are two drawbacks to this — memory and speed. Putting the variable in takes up memory, only one byte but it still takes up a little.

On a '64 this may not matter but on a 3.5K VIC every byte matters. The other drawback is speed. If you omit the variable the computer assumes that it is in the right loop and ploughs on, however if you include the variable it stops for a fraction of a second to check that it is the right variable before proceeding. For this last reason I would omit the letter but this does not mean that your friend is wrong, just less efficient.

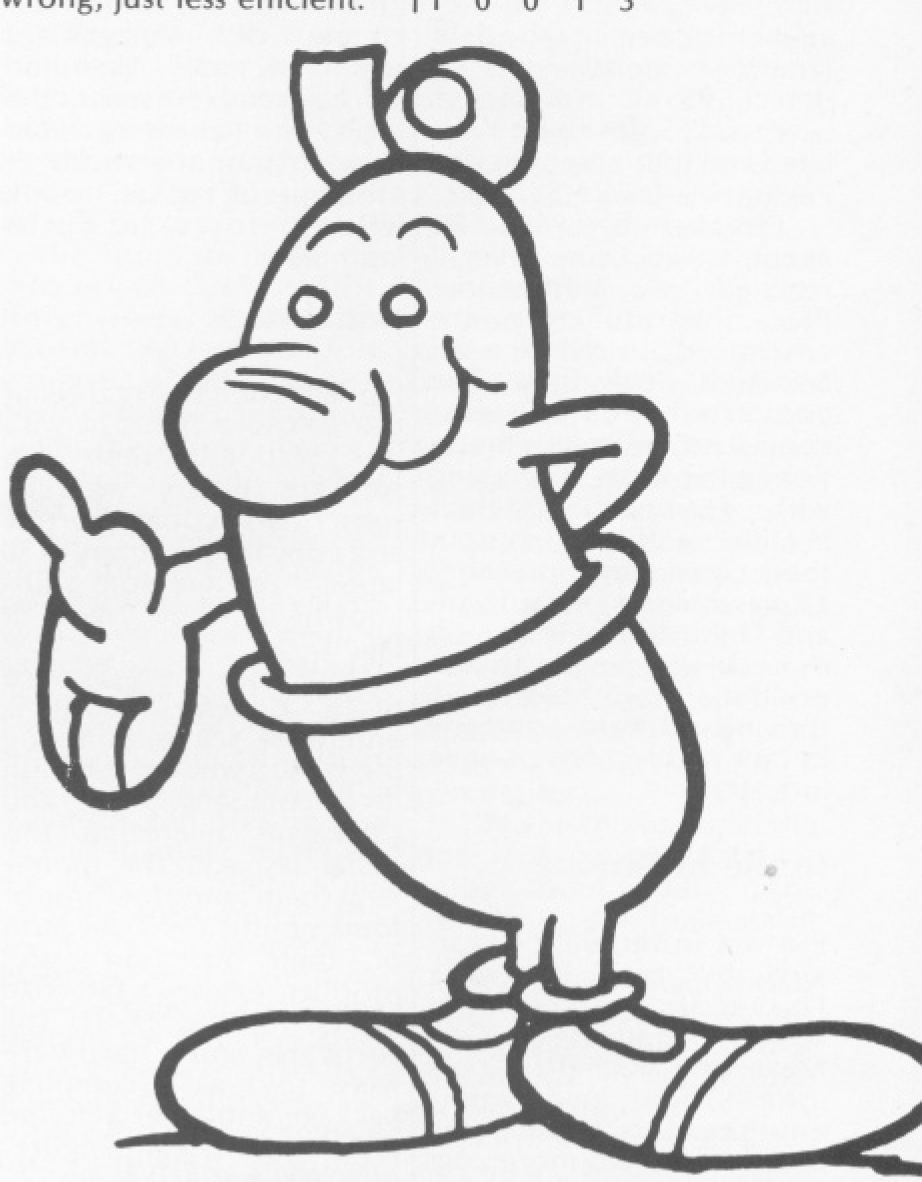


Hint. You can save memory in data statements by not using zeroes. If you just have a comma the READ statement will assume zero, or null for strings:-

```
20 READ N
30 PRINT N,
40 NEXT
50 DATA 1,,,1,3
```

Will produce

```
1  0  0  1  3
```



OUTPUT

BEHIND CLOSED DOORS

Your Commodore's Alison Hjul heads north in search of some insight into the soaring success of Coventry-based software company, PSS.

I WAS SENT TO COVENTRY yesterday. No, my work colleagues hadn't agreed to ostracize me from their company: the Coventry in question was the Warwickshire city of motor industry and modern cathedral fame. My mission — to detect PSS's formula for success. PSS, for the uninitiated is the acronym for Personal Software Services.

Escorted by PSS's PR agent, David Crossweller, I reached my destination. PSS's headquarters are ensconced in a slice of Midlands' suburbia. The converted-semi exterior concealed a conglomeration of offices brimming with examples of their prolific software output (both past and present), assorted pieces of hardware and industrious staff, all more in keeping with the position they hold as a thriving software company in a highly competitive industry.

In the beginning. . .

PSS was instituted $2\frac{1}{2}$ years ago by two Warwick University management science graduates, Gary Mays and Richard Cockayne. These two young entrepreneurs, unwilling to become yet two more cogs in the wheel of a large company, decided to set up their own business.

A quarter-page advertisement for computer games on tape instigated their entry into the world of computers. With a little mental arithmetic and a modicum of common sense, the duo established their business on the criteria that a 50p tape could justifiably be sold for £5. Armed with no computing knowledge but with their couple of management science degrees, Mays and Cockayne advertised for the tools with which to build their dream: the result — a deluge of replies from a willing force of programmers.

Initial steps

PSS launched their career in the software industry with the creation of their own software library whereby subscribers could borrow and copy from a library of 30 tapes. For reasons unknown to Gary Mays, it failed.

But, with the assistance of a Coventry business enterprise scheme offering them £40 a week to establish their company, Mays and Cockayne overcame this initial setback. The money kept them going for three or four months until the birth of their first successful product, QSave, a fast-load device for the ZX-81. The software and hardware were sold as a complete package and, over a period of 18 months, a sales figure of 30,000 was achieved.

Commodore hits the scene

Although PSS now design and write their own software, they originally marketed largely the produce of other software houses. Having concentrated mainly on Spectrum software, by March 1983 they were unable to avoid the advent of Commodore onto the British market. Claims Gary Mays:

"We didn't have a huge success with Spectrum: we came in too late. Everything was ticking over nicely, but it occurred to us that Commodore were going to be big".

The first hurdle they had to cross was the absence of suitable Commodore products. A lot of their software had previously been submitted by youngsters but Gary Mays saw Commodore as a different kettle of fish altogether.

"For some reason Commodore seems to be different in that kids don't program it".

"I went to Chicago for the CES [Consumer Electronic Show] last June with the sole intention of picking up Commodore Software from a stateside company".

Symbol Software came to their rescue and they soon had a hit with Neoclipse which featured in the star list of various computer magazines. Gary Mays attributed Neoclipse's success to its exploitation of Commodore sprites and graphics which many of their rivals appeared to neglect. Also, quite simply, 'It was fun to play".

Birth of a concept

Now that PSS are producing their own software, nearly all their programs are produced by in-house programmers rather than freelancers, which has given them tighter control over both programs and time scales.

"We were ending up with a product which, as far as they [their external programmers] were concerned, was finished". And "We never knew when it [the program] was coming".

How do PSS attain their ideas? Gary Mays again: "The initial spark for an idea can come from anyone. . then Richard and I and Campbell MacCausland, the software manager, sit around a table and thrash ideas out until we agree".

Although hardly in his dotage himself, Gary Mays says that one problem PSS do face is getting onto the same wave-length as the kids (of whom their market is largely composed). So as to escape the danger of swamping the market with their own ideas, PSS include with their packages questionnaire cards to pick the brains and ideas of their buyers. This has proven a very successful formula.

Until recently, their Commodore games have been largely arcade-type games.

"We try to write simply what the market wants". Judging by the direction in which PSS are now heading, their users must be crying out for something more complex than a simple arcade-type game.

Into battle

Midway, a wargame based

on the famous World War II battle, was PSS's first British produced package for the Commodore. Although on release for no more than 3 weeks, at the time of going to press, the reaction to Midway already seems very encouraging to PSS's Gary Mays.

"It was a bit of a gamble. Alan [Alan Steel, Midway's creator] has been a wargamer since the age of 16. He kept getting different wargames but got fed up with the standard. He came to us and suggested he wrote a wargame".

In fact, since a complete wargaming system has now been devised, Midway promises to be the first of a series.



Gary Mays and Richard Cockayne

On your bike

With their newest offering, Hyper Biker, PSS are indeed satisfying the latest craze. Much to Gary Mays' relief, an idea instigated 8 or 9 months ago hasn't waned.

"Because BMX biking seems to be taking off in a solid sort of way, it's maintaining. Practically every kid I see seems to have a BMX bike. . it looks good and applies well to the computer".

PSS's software manager, Campbell MacCausland, gave me a brief overview of the game. It's a 4-player, 3-D game with joystick or keyboard control of the bike. Gary Mays interjected:

"The first thing the player has to get to grips with is controlling the bike. An awful lot of research has gone into getting it realistic and playable".

With a selection of eight different events to choose from, for example, a flat race, an obstacle race or a wheeling competition, coupled with such accurate bike control, Campbell MacCausland believes Hyper Biker has captured as closely as possible, the real thing.

"We've really gone out of our way in not just calling the game BMX but trying to simulate as near as possible what would be done in BMX competition".

With a scrolling display enabling the background to pass by as you pedal, graphics which they claim cannot be faulted and the afore mentioned features incorporated into a game, which, above all, is fun to play, PSS believe they are onto a winner.

Gary Mays' retort to my, perhaps, insolent remark that maybe a child, thrilled by the speed and excitement of riding a BMX bike would not be so enraptured by the prospect of operating a simulated BMX bike through the medium of computer, monitor and joystick, was that there was ". . .no reason why competition in computer games could not be the same as in a street".

Maybe, come December, a TV screen in a fire-lit sitting room will be more enticing than a wet and windy street, especially if Hyper Biker is as realistic as its makers claim.

Magical mystery tour

With their latest brainchild, 'Swords and Sorcery', PSS claim to have surpassed anything yet imagined by themselves or their competitors. In fact, the concept of a computerised version of the role-playing game, 'Dungeons and Dragons', has been swimming around in PSS brains for a long while. The program design has been underway for close on two years now; actual programming commenced 9 months ago and is now nearing its conclusion. Mike Simpson, its creator, is a 'Dungeons and Dragons' 'expert' and a highly competent programmer.

Gary Mays is very proud of his new baby: "Everyone who's seen it said 'You can't do it' ".

"We've tried to make it the ultimate mix of Arcade and Adventure. It'll be the game of the year".

Campbell MacCausland continues: "It makes the Hobbit seem like Pacman".

"The problem we're going to have is making people believe it's as good as it really is".

The product uses the unique MIDAS (Multi Dimensional Animation System) system which, amongst other facilities, provides full 3-D animation, which, PSS believe, makes it as close as you can get to a video disc game.

"Swords and Sorcery' (which should be available mid-October) allows you to develop your own unique character and to experience, in that guise, a series of adventures through assorted underground corridors. Should you tire of one adventure, with the aid of a set of expansive modules, you may transfer your character to another. PSS also hope to provide a networked system on both the Spectrum and Commodore by January, thus adding even further dimensions to the game.

They expect to develop a cult following through 'Swords and Sorcery'.

(I don't believe I can do justice to 'Swords and Sorcery' in the allotted time and space but we hope to review this revolutionary game in a future issue of 'Your Commodore' — so keep your eyes peeled!).

Selling the goods

PSS certainly seem to have their ears to the ground as far as coming up with the right product at the right time is concerned. Where other companies with, seemingly, as much potential have floundered, they have not only survived but have managed to achieve a 100% increase in turnover within the last year. The market has exploded and PSS have kept abreast of it. They also feel that success has enabled them to take more of a gamble; thus, they can follow through ideas which more cautious companies can but dream about. And, naturally, the higher the standards they set, the higher the standards they will be expected to attain and, thus, will endeavour to maintain.

Advertising and the assistance of a PR company are also quite indispensable in a fast and furious business which Gary Mays likens to the record singles' market.

But PSS are blooming under such pressure. When questioned on his views on the industry's future, Gary once again borrowed the record analogy by suggesting 'albums" of software. These, he believes, would work in everyone's favour by extending the life of a piece of software.

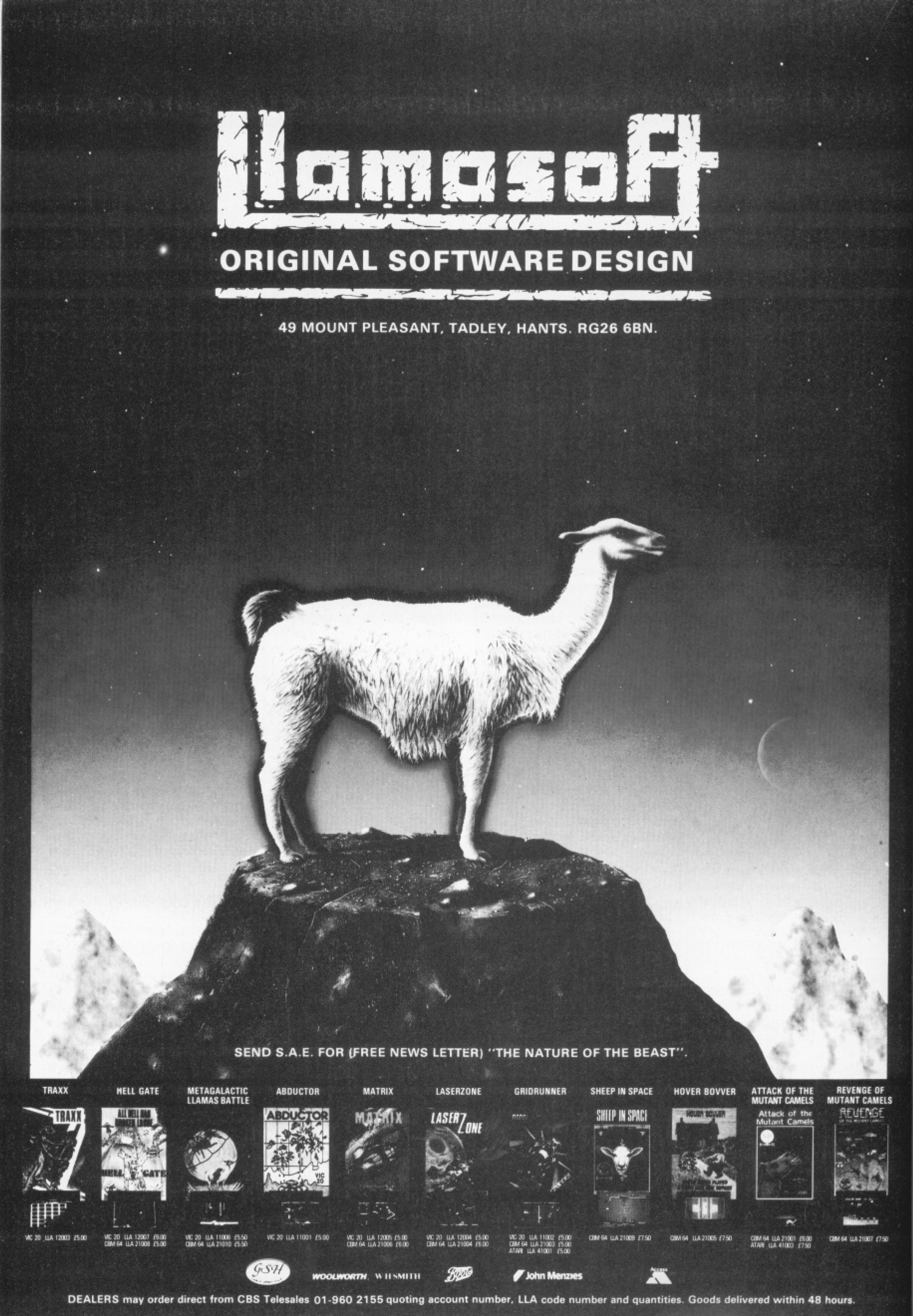
"I think what we'll see are compilation tapes or discs". But such a concept seems alien to PSS's competitors.

"At the moment, on the one hand people talk together about issues, on the other they don't talk about money".

And so to their future with Commodore: does Gary Mays see PSS opting increasingly for Commodore products?

"I think we've got to: it's a world market rather than a UK one."

With their acute insight into the software industry and courage to pursue a novel concept, I hope that PSS do maintain their confidence in Commodore as a vehicle for their products.


Llamasoft
ORIGINAL SOFTWARE DESIGN
49 MOUNT PLEASANT, TADLEY, HANTS. RG26 6BN.
SEND S.A.E. FOR (FREE NEWS LETTER) "THE NATURE OF THE BEAST".
TRAXX
VIC 20 LLA 12003 £5.00
HELL GATE
VIC 20 LLA 12007 £6.00
CBM 64 LLA 21008 £5.00
METAGALACTIC LLAMAS BATTLE
VIC 20 LLA 11006 £5.50
CBM 64 LLA 21010 £5.50
ABDUCTOR
VIC 20 LLA 11001 £5.00
MATRIX
VIC 20 LLA 12005 £5.00
CBM 64 LLA 21006 £6.00
LASERZONE
VIC 20 LLA 12004 £5.00
CBM 64 LLA 21004 £6.00
GRIDRUNNER
VIC 20 LLA 11002 £5.00
CBM 64 LLA 21003 £5.00
ATARI LLA 41001 £5.00
SHEEP IN SPACE
CBM 64 LLA 21009 £7.50
HOVER BOVVER
CBM 64 LLA 21005 £7.50
ATTACK OF THE MUTANT CAMELS
CBM 64 LLA 21001 £6.00
ATARI LLA 41003 £7.50
REVENGE OF MUTANT CAMELS
CBM 64 LLA 21007 £7.50
GSH
WOOLWORTH
W H SMITH
Boots
John Menzies
Access
DEALERS may order direct from CBS Telesales 01-960 2155 quoting account number, LLA code number and quantities. Goods delivered within 48 hours.

DOING IT YOURSELF

Dissatisfied with off-the-shelf business systems? Think you can beat the 'experts' at their own game? Grahame Davies offers sound advice on writing your own bespoke business software in the second installment of this series.

LAST MONTH I SAID ALL my routines would sit between lines 1000 and 2999. The first routine that you must do will be an exception to this. This is perhaps the most important routine — it is the one which saves your program to disc giving version numbers and maintaining three (or more) backups. As you develop your program, you will undoubtedly save it onto more than one disc. You may later find yourself, with two discs with the same program but one will be a later version than the other — so how do you find out? — what was the latest change to that program? — probably you changed part of one line to fix a bug but which line? Another thing to remember is that it is a good idea to save your program before running it. The reason for this is that it is surprisingly easy to get a BASIC program to hang the computer. Common ways of this happening include using POKE without setting variables or using the wrong ones. This will almost certainly result in memory location zero being POKEd with a random number thus repointing the 64's memory around and pageing out BASIC (unless you are lucky). Finally there are obvious advantages to having three or more backups available — disc corruptions being the first which spring to mind!

No to the code. I have put it at the end of the program so that it is out of the way. The line before it starts is an END instruction in case your program does not stop before it gets to this routine. The routine will keep a version number at the start of the BASIC program so the start of your program will look like this:

```
10    rem"Program or
      System Name
20    rem"Version 00
```

Note that in line ten you must have the closing quotes and the number of characters between the quotes exactly correct because we are going to be altering the version number which must be a fixed number of bytes from the start of the program. There must be 32 characters between the quotes. On line 20, the positioning of the version number is equally as important.

The routine checks that lines 10 and 20 have been entered correctly and gets the current version number (version number = b*10+c). In line 60080, set f$ to your program name (make it less than 15 characters though as the version number will be added on to this). Line 60100 opens the error channel to the disc and, if we are not up to version five, jumps over the next bit of code which scratches the oldest version from the disc. The current version on disc is then renamed giving it the existing version number then the version number in this program is altered and this program is saved under the name in f$.

If you type this program in and keep f$ as "system", then run 60000 five times you will end up with the latest version on the disc being called "system" and the backup versions called "system02", "system03", etc. so you know where your latest version is.

```
59999  end
60000  open1,8,15:close1        :rem abort disc files
60020  s=2048:a=s+55            :rem s= start of program
60030  ifpeek(s+39)<> 34thenprint"missing quote on
       line 10":stop
60040  b=peek(a)and15:c=peek(a+1)and15:rem version
       no.
60060  ifb>9orc>9thenprint"illegal version no.":stop
60080  vn=b*10+c:f$="system" :rem version : system name
60100  open1,8,15:infvn<5then60200
60120  a$=f$+mid$(str$(100+vn-4),3)
60130  print"scratching              "a$
60140  print£1,"s0:"a$:gosub6100
60200  a$=f$+mid$(str$(100+vn-1),3)
60210  print"archiving last version"a$
60220  print£1,"r0:"a$"="f$:gosub61000
60300  pokea,int(vn/10)or48:pokea+1,(vn-int(vn/10))or48
60320  print"saving "a$
60340  save(f$),8:gosub61000
60400  end
61000  get£1,a$:printa$;:ifa$>chr$(13)then61000
61020  return                            :rem disc status print
```

The subroutine at 61000 reads the error channel from disc until a carriage return is reached, printing it as it goes along. The reason for doing it this way rather than using input£ is that this formats it neatly and it is unnecessary to check the value of the status except after you have saved the file.

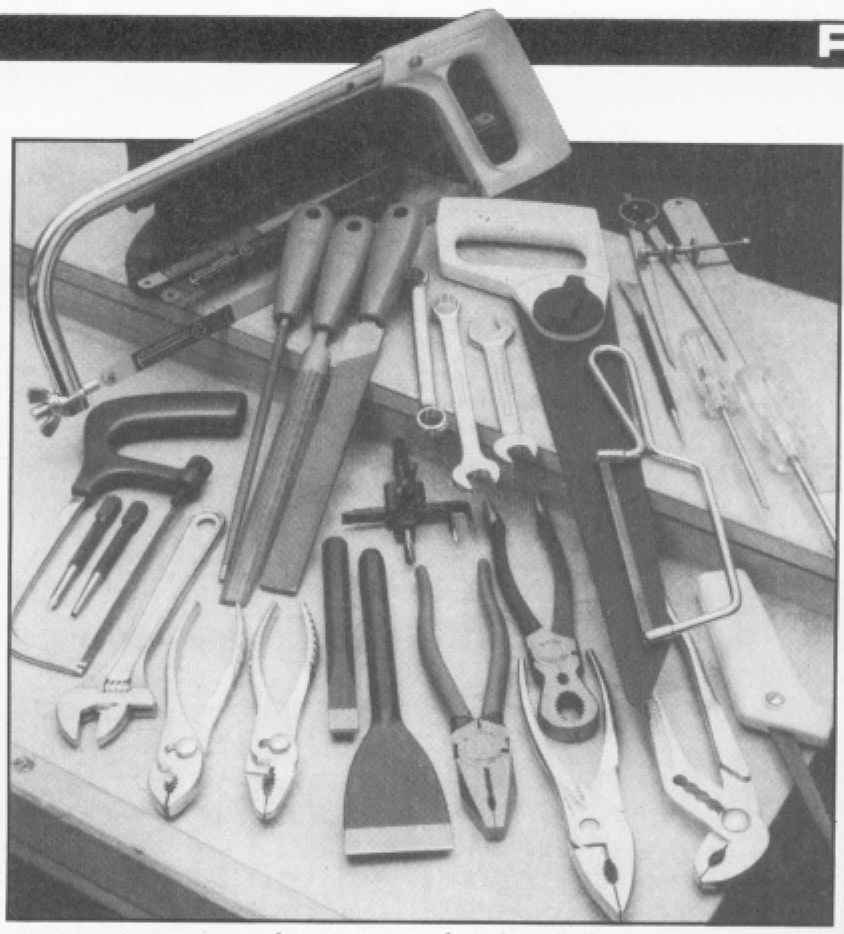
Defining standard strings and constants

At the start of your program you will need some standard strings and constants that you will often use in your programs. You will of course add all your own ones to this brief list.

```
1000   fori=lto80:sp$=sp$+""
        :next:rem set up
        space$ of 80
1020   q$=chr$(13):qu$=chr
       $(34):ec$+chr$(27)
1030   rem carriage return:
       quotes:escape
       character
1040   bo=13*4096+32:ba=
       bo+1   :rem border
       :background
1060   sn$="Record
       Keeping"   :rem
       system name
1080   fori=0to21:poke13*
       4096+4*256,0:next
1090   rem clear out sound
       chip
1100   dim the arrays you
       need here
1120   define any functions
       required here
```

The above is quite self explanatory but you may wonder why line 1020 is included. A carriage return is often required when printing to a printer or a disc drive and it is a very useful way of truncating data on

input. If used regularly, it is quicker and easier to type q$ than chr$(13). The quotes (chr$(34) cannot be typed into a string so if it is required, you will have to type chr$(34). Once again, it is easier to type qu$ and far more reliable then trying to remember its code every-time. The last one, the escape code, may not be needed. If, however, you are talking to an RS232 device such as a printer then the escape code is essential as the listening device will perform special functions depending on the data received after the escape. I have included it for completeness and as a further example of the kind of thing you will find useful to set up.



Key to success

No matter how fast, fanciful, clever and flexible a program is, the major key to its success is how pleasant and easy to use it is. This entails making the information on the screen and printer appear in a neat, formatted style. It also means that the input of the data wants to be safe and friendly. The BASIC input command is provided of course but it falls far short of the necessary standards and flexibility required. It does allow full screen editing but does not trap any characters such as home or clr home, cursor up, cursor down and so on. It is evident that we need a flexible but safe utility to do this. We need to specify where on the screen we want to input, how long the field is, what sort of data is to be typed into this field and so on. After all, you do not want accidental key pushes to cause the program to crash or wrong data to be entered. For example, if we were asking for a number, and we accidentally type 1e45 instead of 145 (not unlikely due to the proximity of the keys) we would get ?overflow error when using the basic input command. Even if we entered it as a string then took the val of it we would get the same error. This is more likely than you may think as I know of one package which when inputting data, took the val of every string regardless and then produced this error in the middle of an address! This was because the post code was something like 5e4 5ds.

This subroutine will require several variables passed to it. You will find that you can easily improve the flexibility of this subroutine to include your own special function keys. You could have one which aborts the whole screen or takes you to the bottom entry and so on but each of these keys will have to be tested for separately in your code. This routine will require the starting position across the screen to be in ac%, the starting position down the screen to be in do%, 1% will be the length of the field, ty% will be the type of the field and the default will be in df$. The default is what was in the field before you started editing. If creating something, then it will be zero blank most likely but, if amending something, it should contain the details from the file. The type will be one of the 3 below.

Routine A is quite straightfoward but is very flexible. Once you are happy that you understand it, try adding a function which clears everything to the right of the cursor. At the moment, the only way out of the routine is to press return but we can add to this. You may use the function keys as when they are pressed you can test the ASCII of them. I suggest you make this routine return another variable such as fk to say if a function key has been pressed and if so, which one it is. Do not forget to set fk to zero if return was pressed though.

The 3 types of variable available.

1 = any alphanumeric character (excluding comma & colon)
2 = positive or negative floating point number
3 = positive or negative integer

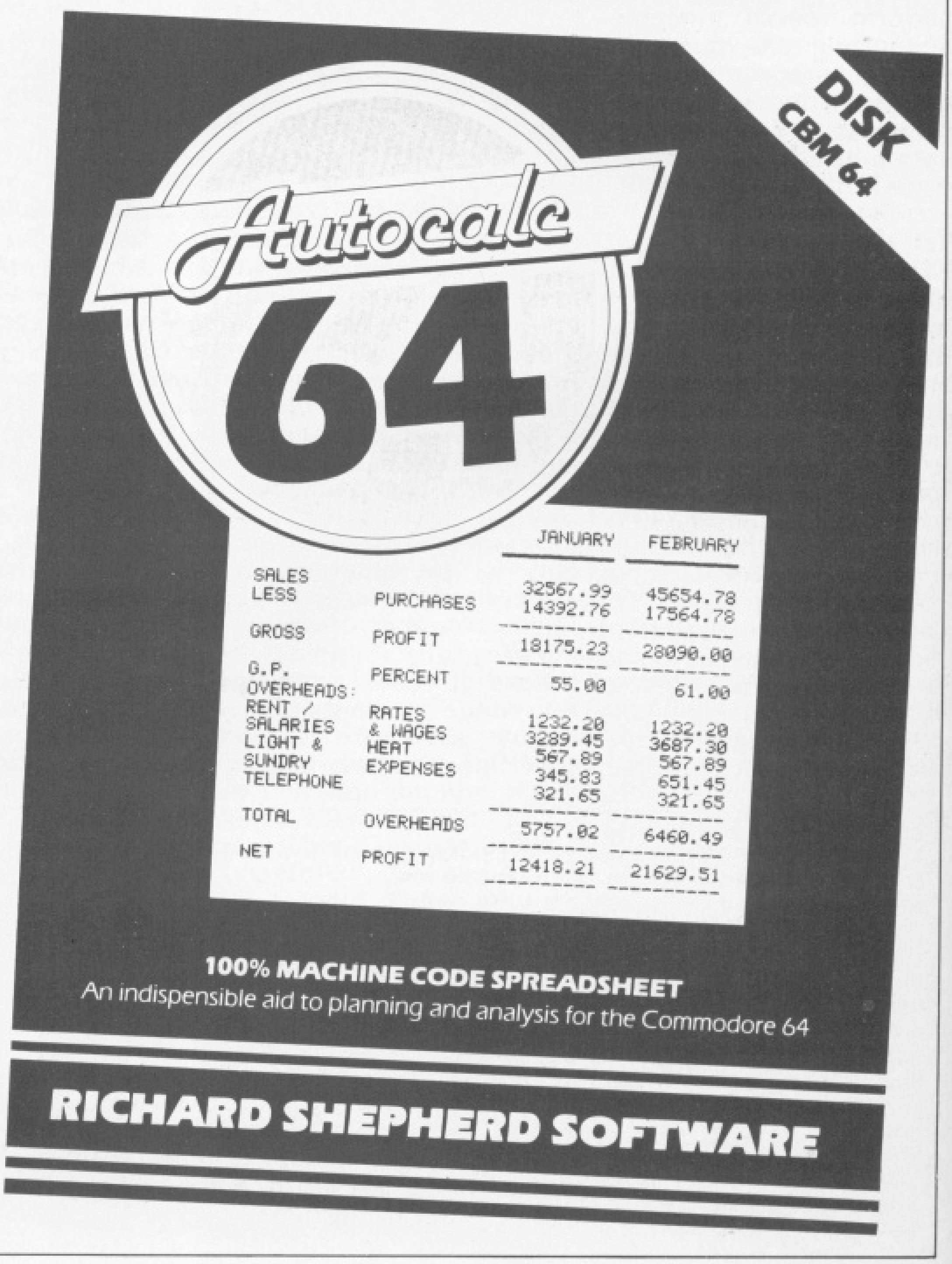
Once you have got this routine to work, try adding other types such as positive only integer and so on. The routine uses b$ to store the new data in and returns 'a' as the value of the field (if numeric).

```
1140  do$= <home> +cursor down * 25
1150  rem set up do$,sp$,q$ and qu$ at the start
1200  b$=left$(df$+sp$,1%):a=1:rem make b$ correct
      length
1210  printleft$(do$,do%)tab(ac%–1)left$(b$,a–1);
1215  print <rvs on> mid$(b$,a,1) <rvs off> mid$(b$,a+1)
1218  rem a is the cursor position along b$
1219  rem  <rvs on>  on is quotes, control 9, quotes
1220  geta$:ifa=“”then1220
1230  ifa$=chr$(29)anda < 1%thena=a+1:goto1210:
      remcrsr right
1240  ifa$=chr$(157)anda > 1thena=a–1:goto1210:rem left
1250  ifa$=chr$(148)andright$(b$,1)=“ ”then b$=left$
      (b$,a–1)+” “+mid$(b$,a,1%–a):goto1210
1255  rem line 1250 is insert
1260  ifa$=chr$(20)anda > 1then b$=left$(b$,a–2)+mid$
      (b$,a)+“”:a=a–1goto1210
1265  rem line 1260 is delete
1270  ifa$=q$then1395                    :rem pressed return
1320  onty%goto1330,1340,1350,1360,1370:rem each type
1330  if(asc(a$)and127) >31anda$< >“,”anda$< >“:”
      and a$< >qu$then1380                       :rem type 1
1335  goto1390
1340  ifa$=“.”ora$=“–”or(a$ >=“0”anda$ <=“9”)then1380
1345  goto1390
1350  if(a$ >=“0”anda$ <=“9”)ora$=“–”then1380
1355  goto1390
1360  rem add in type 4 here
1370  rem add in type 5 here
1380  b$=left$(b$,a–1)+a$+mid$(b$,a+1,1–a):a=a–(a < 1%)
1385  goto1210:rem char ok: inc pos in string until end
1390  goto1210:rem add in an error beep here
1359  printleft$(do$,do%)tab(ac%–1)b$:ifty > 1thena=val
      (b$)
1398  return                                      :rem exit
```

Routine A.

BUSINESS FILE

Our business expert, David Crisp, assesses some of the latest business software releases for the Commodore.



THIS MONTH I'VE BEEN FEEDING my 64 with some fairly low-cost software which aims to help you work out your business or home finances. The first one I tried was **Autocalc 64** from **Richard Shepherd Software.** It costs £14.95 on cassette and £19.95 on disc and is a low cost spreadsheet. After loading you are asked whether you are using tape or disc. This is fine to start with but, after having to specify tape or disc more than a couple of times, it becomes very much a chore. If you have never used a spreadsheet before I feel that this one may put you off them. A good spreadsheet is an invaluable aid for financial planning and financial analysis but they are unforgiving things and can be very frustrating.

This spreadsheet does all the important things that spreadsheets should do — the frustrating thing is, how it does it. When moving from one part of the sheet to another there is a very disturbing screen flash which tires the eyes after a while, and with a machine with the capabilities of the 64 there are much smoother ways of performing a sideways screen scroll.

Perhaps the most disturbing thing about it was when it crashed. I had spent about an hour copying in a set of information and calculations and intentionally put in a division by zero. Instead of the expected error message the whole thing crashed. On trying again with less information it performed correctly and then on a third run it crashed again. Disturbing.

There is no printer interface software built in and my software would not run at the same time as this and so I was unable to test the printout facility. A lot of people now have Centronics printers and this program is limited to just a few of Commodore's own and a couple of others. Use others and you invalidate the guarantee.

The new documentation is barely adequate for a spreadsheet program and only describes the bare bones of what to do. I get the impression that everything is being left to the demonstration files.

I was very disappointed with this program and with its lack of documentation and inability to give a printout on a good range of printers. I would say that it is only really suitable for people who want to fiddle with a spreadsheet. If you really want a spreadsheet to use in your business then I recommend that you spend a little more money and get something with more potential. It is true however that it is a low price program and because of this I feel that it is reasonable value for money. Shame about the crashes.

After a couple of hours on that one I loaded up **Figaro** from **Saxon Computing**. The blurb on the back was mouthwatering and I couldn't wait to get it into the machine. I had

to restrain myself from diving straight in and so left the computer to spend an hour with the manual. I'm glad I did. The amount of information was incredible.

The program is a type of database. It is intended to store numeric information and analyse it in different ways presenting final output as a list of comparative information or in virtually any type of graph you care to mention. Because of the complexity of the program I only had time to work on the demo files provided — and these impressed me. I feel that this is a program that could prove invaluable to any business where cash projections, growth rates, sales targets, and seasonal forecasting etc is invaluable. I have doubts as to whether many small business users would be able to stretch it to its limits but if there are any financial wizards out there who want to impress the boss then this is a winner.

Now the bad news. It says that high resolution printouts are possible using Commodore and Centronics printers. I tried with a CP 80 and a Smith Corona TP1 and got nothing. Both printers checked out OK and work well with Superbase and Easy Script etc. We'll be checking with Saxon to try and discover why things did not work; I hope I will be able to tell you about the printout facility in a later issue.

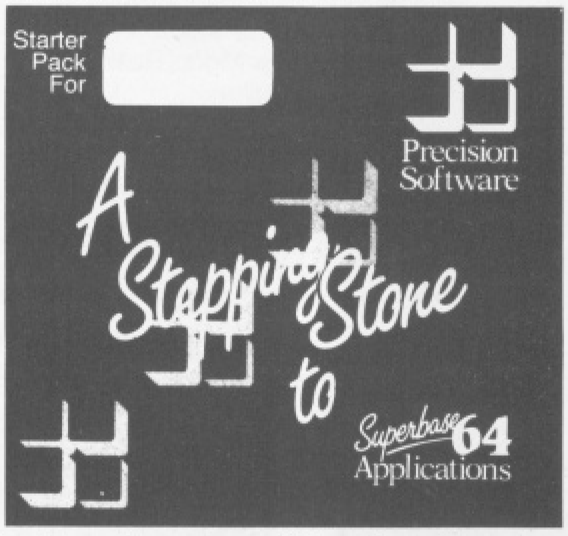
In an article like this it is not possible to describe its potential fully as it would need a whole article of its own. That may be possible in the future. In the meantime, if you feel that this is something you may be able to use, I recommend you pop down to the local computer store and have a look. I think you will be impressed. Damn shame about the printout. . . .

Next day now, and I have just loaded up **Purchase Ledger** from **Kemp**. I use a purchase ledger program in my business and it was my intention to run this in parallel with my existing system. Somebody who knows a lot about purchase ledgers has written this program. Unfortunately, they don't seem to be totally effective programmers.

This had a lot of potential as all the functions were there. It was let down by its poor display, non-existent error trapping and inconsistent inputs. Some parts of the program require you to input 'yes' as a whole word, other times just 'y' will do. On the main option page, if you make an incorrect entry, up scrolls the screen and eventually the whole menu disappears until you have to guess what the menu said. During stages of the program, one touch of the break key will halt everything and typing in CONT only works sometimes. CAN'T CONTINUE ERROR comes up 50% of the time. Re-RUN and it's goodbye to your data.

Kemp manuals are always quite good but simply warning you not to touch the CLR/HOME key and then saying, if you do, just reposition the cursor, is not good enough in a business program. That type of thing should be error trapped and Run/Stop keys should be displayed. It can be argued that a program like this is easy to tailor to your own needs, but specific entry points can be put into a program to enable you to tailor a program.

Needless to say, I did not run this in parallel with my existing ledger as, in short, it was just not up to the job. Sorry.



Ten Superbase Stepping Stones piled in front of me. Great I thought as I am a Superbase fanatic. The programs can only be run with Superbase and the titles I had for review were as follows:

Club Records
Estate Agents
Job Costing
Purchase Day Book
Cash Book
Accountants Time Recording
Solicitors Time Recording
Stock Records
Travel Agents
Sales Day Book

Some of these I would not know enough about to give a valid review so I will only be able to give an overview of what they were about. It must be pointed out that these are *just* stepping stones. They are not complete applications. Each one consists of ready formatted records and pre-written report layouts. This means that the easy bit is done — the hard part, which is linking it up to an actual Superbase program, is not. To be fair, it is possible to use them from a menu and so they are ready to go in that respect but it is a long winded way to go about it. They are excellent for showing the potential of Superbases and can easily be modified but, as I say, it is not a complete and fully running interlinked application.

To pick one out and show the sort of thing they do was a hard job but in the end I went to the Club Record Controller. This I feel is one that most people may be able to use. The files set up are orientated towards the sports type of club not the stamp collector type but this could be modified. It will produce a members list, overdue fees, equipment on hire, what each person's particular interests are, all the things in fact that a club secretary would need to know.

If you have or indeed are thinking of buying Superbase (which I recommend) and are going to use it for an application for which a **Stepping Stone** is available then it would be a good buy. It would take a lot of the groundwork out of writing the finished article and would probably point out some little things that are easily forgotten. Documentation with the Stepping Stones is non-existent but you can print out the HELP pages to produce a fairly comprehensive DIY instruction manual. All in all for the money they cost they are a good buy and they are something which I shall use in the future. NICE WORK.

Back to **Richard Shepherd** and his **Cash Controller**. Basically this is a home budgeting system which performs quite well. You put in the amount you intend to spend on such things as phone etc etc and, as time goes on, it works out for you if you are under or over budget. (My bank manager usually does that anyway). With this you can also keep track of your bank account and it will take into consideration standing orders, etc, and will produce a statement on demand. To me the most useful part was a **loan/mortgage** calculator. I certainly learned a thing or two on that part. If a particular company gives you a quote for credit it will work out a fourth variable from three you must enter, eg, if you borrow £1000.000 for 12 months at 21% APR it will tell you how much you must actually pay back. Quite shocking some of them. A home budget program would not be for me but I am sure the loan/mortgage calculator will save me more than the program costs. If you are looking for this type of program then I feel that you could not go far wrong with this one.



The Prizes:

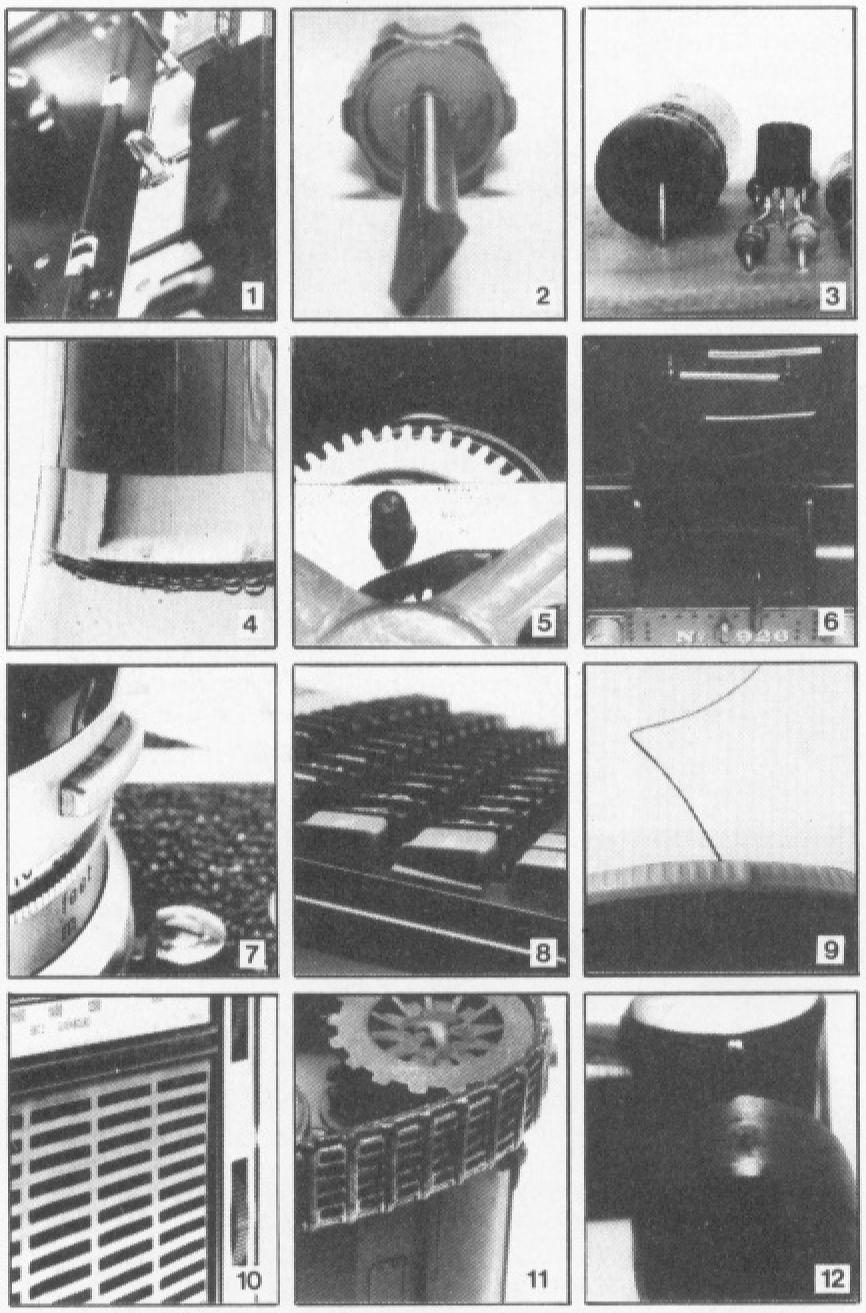
You could win £2,500 to be spent on a dream holiday of your choice for you and your family!

Second prize — a complete Canon portable video outfit worth £1,300.

Third prize — a BBC Model B micro computer plus software worth £450.

Fourth prize — Minolta X700 camera with a 50mm lens and flashgun, worth £280.





How to enter:

Just identify the twelve objects pictured opposite....

HINT — the Argus Specialist Magazines listed below might give you a clue:

Electronics Today International
Personal Computing Today
Movie Maker
Your Model Railway
Clocks
Home Computing Weekly
Beatbox
Ham Radio Today
Electronics
35mm Photography
Model Cars
Woodworker
Games Computing
Photoplay Movies and Video
ZX Computing
Military Modelling
Hi-Fi Now!
Winemaker
Citizens' Band
Model Boats
Video Today
Popular Crafts
Which Video?
Your Commodore

and write your (one-word) answers in the spaces provided on the coupon. For instance, if you think that number 9 is a record, write 'record' in the space next to 9 on the coupon and so on. Then tell us in up to 20 words why **MAGAZINES MAKE IDEAL HOLIDAY READING.** Complete the coupon in BLOCK LETTERS, and send it to: *DREAM HOLIDAY COMPETITION, Argus Specialist Publications Ltd., No 1 Golden Square, London W1R 3AB,* to reach us no later than 31st December 1984.

Competition rules

1 The competition is open to all UK and Eire readers except employees of Argus Specialist Publications Ltd., their printers and distributors.
2 As long as an original coupon from the magazine(s) of your choice is used for each entry there is no limit to the number of entries per person. Photocopied coupons will not be accepted.
3 All entries must be postmarked before 31st December 1984.
4 The prizes will be awarded to the first four entrants who identify the twelve objects correctly and whose completed sentence is judged the most apt and original.
5 No correspondence will be entered into about the competition results; the judges' decision is final.
6 Winners will be notified by post and the results will be published in a future issue of this magazine.

The 12 objects are

1. .... 2. .... 3. ....
4. .... 5. .... 6. ....
7. .... 8. .... 9. ....
10. .... 11. .... 12. ....

Magazines make ideal holiday reading because *(up to 20 words)* ....
....

NAME *(BLOCK LETTERS)* ______________ AGE *(if under 18)* ______

ADDRESS ______________

Send to **DREAM HOLIDAY COMPETITION,** Argus Specialist Publications, No 1 Golden Square, London W1R 3AB.

The SX64, a portable version of the Commodore 64, has been freely available for over six months and seems to have established a solid base. David Crisp weighs up the pros and cons of Commodore's portable micro.

THE SX64 REVIEWED

The Commodore SX64 is a Business/Home micro which has a built in 5" colour monitor and a built in 1541 single disc drive. The DX64 was intended to be a dual drive version but according to Commodore it will not now be released. I am led to believe that overheating of the second drive is the reason for this. However, an ordinary 1541 drive can still be connected up through the serial port. It comes with some free software which I will deal with later and a manual which is a rehashed version of the regular 64 book with parts of the 1541 disc drive handbook mashed in. As with most Commodore documentation, this is a disaster and in many places totally inaccurate. Some work has been put into correcting errors that were contained in original manuals but just as many new mistakes seem to have crept in.

The SX64 looks good. It is dark grey and some thought seems to have been put into making it aesthetically pleasing. There is a very thick, robust handle which makes carrying the moderately heavy machine much easier and no protruding parts on which to catch your legs. The front of the machine, when opened, reveals a slightly smaller keyboard than the normal 64 with dished white keys with the graphics symbols clearly marked. The keyboard has a slightly empty feel but over all is comfortable in use although it is quite noisy. The keyboard section is very light and due to that mine has crashed to the floor on many occasions pulling its lead from the socket but so far it shows no signs of damage. The lead which connects the keyboard to the machine can be detached but I find it does not get in the way even when it is carried around so I leave mine in place all the time.



What a view

On the left of the machine is the five inch monitor. Despite being so small the picture is surprisingly clear and I have no trouble reading text providing there is good colour separation. There is slight pin cushion distortion at the edges of the screen but not enough to have any detrimental effect on clarity. The brightness, contrast, colour, vertical hold and volume controls are under a small panel on the right of the machine. They are a little fiddly but, once set, they do not waver far, so are quite adequate. I once read a review where the writer complained that there was no tuning control but on a colour monitor a tuner is not required as there is no DRIFT as on a normal TV set. Also, hiding behind this flap is a reset button. This is not a reset button as on most machines; it serves only to reset the disc drive in case of foul up. If you have a second drive plugged in and the device number has been set through software, pressing the reset button will mean that the drive goes back to device 8. A nuisance if in the middle of a program. The disc drive is very obviously much smaller



than the normal 1541 drive and, to my mind, much quieter as well but alas still as painfully slow. Still at the front end, the most obvious thing is the large black hole just above the drive. In the original this was meant to house the second drive but is now designated as a storage slot. Commodore say in the manual that it is not advisable to store discs in the slot; they do not suggest what you can store in there and so the slot remains relatively useless. Cigarettes and matches live in mine. Seriously I have always kept my discs in the slot and although I would not say this is a safe practice, so far I have lost no data. Suggestions please on what can be kept in this slot.

Weighty problem

This is a portable machine but as I have said it is reasonably heavy. Unfortunately it cannot be used on a train or in a car etc, as it is dependent on a 240 V mains power supply so it is only portable in the sense that it can be taken from place to place, but not used on the way. In use it is identical to the Commodore 64 except when it comes to tape use. There is no facility at all to use tape: in fact the routines that handle tape in the ROM have been savagely torn out. I say savagely because that is all that was done. It would have been sensible when removing tape functions from the ROM to have made all commands default to the drive. This wasn't done and so it is still necessary to suffix all LOAD/SAVE commands with ',8'. If you attempt to use tape, eg with LOAD "progname" ,1, then you get the response 'ILLEGAL DEVICE NUMBER'. Many people say that you do not need tape on a business machine, but it is often forgotten that this is not only a business machine and anyway hasn't anybody heard of executive games? As time goes on though and more and more software is being released on disc, the lack of a tape port is getting less and less of a nuisance.

The compiler for the Commodore 64 will not run on the SX64 and as far as I know this is the only program that will not. If anybody knows of other programs that will not run please write in and say as it may be possible to print a list of those programs and so save other SX users time and money.

In/Out. . .

On the top of the machine is the cartridge port. This is a nice place to have it as it is easy to see and easy to get at. No fumbling at the back of the machine trying to pull out International Football only to find you have also pulled out the disc drive, TV lead and printer. It is difficult not to notice that a cartridge has been left in but should you leave one in and carry the machine then you are likely to take a reasonable sized chunk out of your knee. At the back of the machine are two joystick ports. These are for games and, I presume, the MOUSE when, and if, it is released. Close by also at the back is the serial socket. It is possible to plug in either a second disc drive here or a printer. I have heard some people say that you can only have one or the other attached at once but if you have a second drive plugged in then you can plug the printer into the second drive, a process known as DAISY CHAINING. There is also the versatile USER PORT into which you can plug all manner of things. Centro-

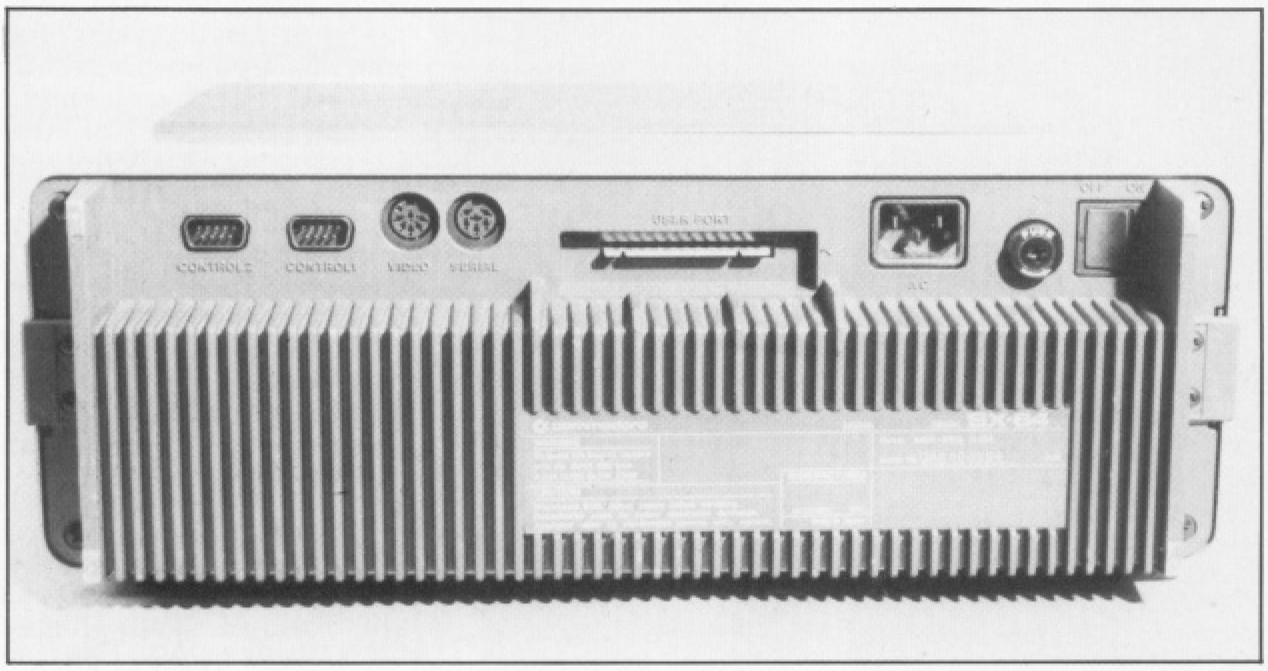
nics printers, IEEE interfaces, in the U.S.A. even a polygraph (lie detector), I believe. The mains socket is also here and finally the audio/video socket. This last one is very important. It has been written that, because there is no modulated TV out socket, it is not possible to plug your SX64 into anything but a Composite monitor. Almost true. For those of you who are lucky enough to have an SX64 and a Video recorder then here is what you do. Your video recorder has probably got an aux.vid in socket. Simply take the video signal from the socket of the 64 into the recorder switch from tuner to AUX and hey presto! a 26" picture. The other advantage of this is the fact that the socket on the recorder is usually at the front of the machine and so there is no swapping of leads behind the television required.

With the SX64 the great thing is the lack of the spaghetti of wires connecting drives to computer, printer to drives etc. This means that you can usually get going by simply connecting the mains and turning on the power with the large easy to find, difficult to hit accidentally, rocker switch.

In use. . .

The SX64 is a pleasure to use. Everything is easy to get at and the screen can be tilted up by using the carrying handle. Because everything is compact there is plenty of room on a computer desk for printers, second drives, books etc and, because the keyboard is remote, it is easy to push it to one side while making notes or reading manuals and so on. I use my machine almost every day to program, to review software, and also to run my business.

From a programming point of view the BASIC is a little dated. There is no easy way to program sound and graphics without using a lot of POKE commands. This is a nuisance but with the many utility- programs these features become easy to use. Sound and graphics on the 64s are excellent. The SID chip controls sound and it is a chip that many synthesisers would be proud to have. There are four sound channels including white noise and these allow stunning sound effects and tunes. The graphics are really something, if you get a chance look at International Football: that should show you how much potential there is in the 64s. There are a lack of disc handling commands and loading a directory of a disc wipes out anything you may have in memory (if not using special routines which add disc commands). Channels have to be opened and closed manually and, although this makes programming a little long winded, I feel that at least I can take a BASIC program and load it into my Commodore 8096 without needing to modify it. There are prices to be paid for compatibility but I feel that some things are worth the sacrifice. Look how much the Electron had to sacrifice to be compatible with the BBC B and eat your hearts out all the Spectrum/QL owners, (does QL stand for Quite Late). There is not 64K of user RAM on the 64 although there is a good chunk of it available. Some extra RAM is also available for machine code routines which is not available for BASIC programs. Because Commodore peripherals are intelligent you will find that, when you plug in a disc drive for instance, no great chunks of RAM are used up to control it. Each 1541 drive has its own on board RAM and a 6502 processor to run it. In the 64 there is what is known as a kernal and this is a boon to machine code programmers; there is not room in this review to describe it fully but in simple terms it is a jump table which allows some compatibility between machines when writing code routines which need to jump to specified ROM routines.



If you are thinking about getting a 64 and a portable machine would be of use to you then I would not hesitate in suggesting that you have a good look at the SX64. It has got its faults but over all I feel Commodore did a good job and, although it is expensive for what it is, I would not be without mine.

SX-tras

When you buy your SX64 you will get some free bits and pieces. Some of it will be software. When I bought my SX64 I pulled out EASY SCRIPT, EASYFILE, FUTURE FINANCE, HIGH FLYER,





COMPENDIUM OF GAMES and a DEMO DISC. That is not quite the truth as when I opened the boxes which should have contained EASYFILE and SUPERSCRIPT a pre-printed note fell out saying "Because of shortages you will find that the discs containing the programs are not here. If you fill in the enclosed form we will send you the discs as soon as they are available.". Well! considering that Commodore do their own disc copying I would rather they had spent time copying discs than spent time printing leaflets saying they did not have your advertised free software. Apart from that, despite sending the forms and phoning up leaving messages on the answerphone and talking to nice ladies, I still have not got EASYSCRIPT or EASYFILE (Please Commodore?). I have got the boxes and the documentation, I only lack the programs. The software that I have been able to look at can only be described as fair but as it is free I am not moaning. High Flyer is a low level business simulation where you have to run your own aircraft business and the compendium of games contains half a dozen games that are being sold off in their magazine as cheapies. The demo disc I like. Apart from some awful spelling mistakes the demos are very good and I'm looking forward to Christmas when I can have the all singing and dancing Christmas card running 24 hours a day. Finally, there is a cloth bag with a large velcro pad in which you can store odds and ends such as manuals leads and floppies. It is big enough to hold 2 standard sized disc boxes and is a very useful thing. A nice touch.

To prove the portability of this machine this review has been written in bed, in my computer room, at work and in the kitchen. THAT' PORTABLE!!!



Specifications

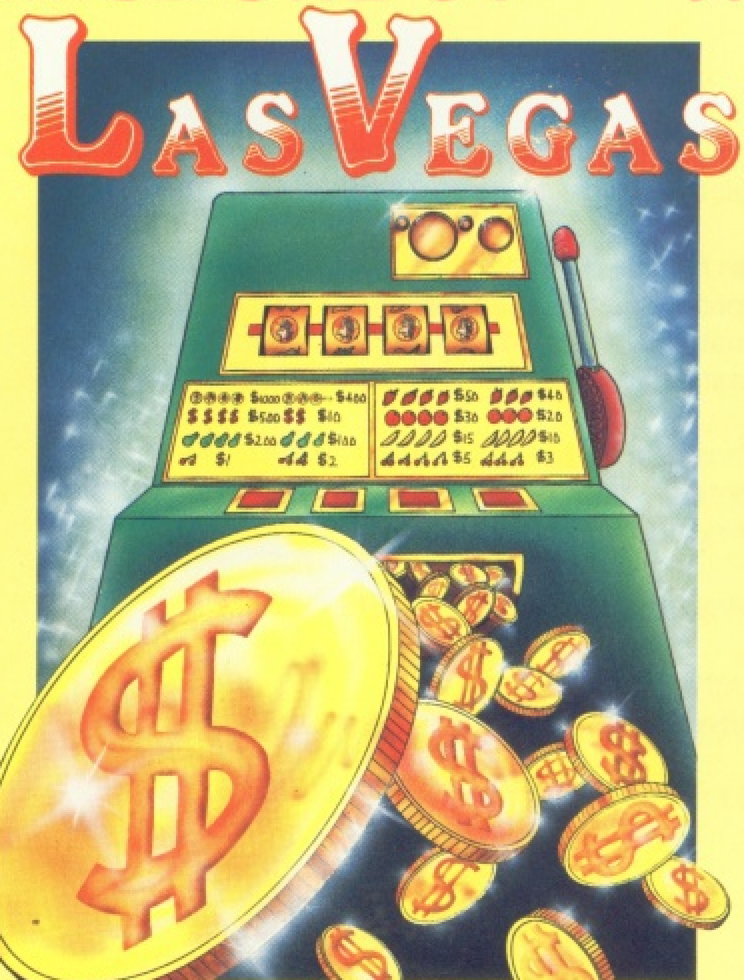
PROCESSOR	6502 (Z80A) as add on
RAM	64K
ROM	20K
I/O	Commodore serial bus; external composite video
AUDIO	6581 SID CHIP
VIDEO	6567 VIC CHIP
LANGUAGE	BASIC V 2.0 64 compatible, with C/PM as add-on
DISC DRIVE:	
PROCESSOR	6502
RAM	2K
ROM	16K
DRIVE CAPACITY	170K
DISCS	5.25"
MONITOR:	
SCREEN SIZE	5"
CHARACTER	40x25
DIMENSIONS:	
SIZE	125 h x 370 w x 370 d
WEIGHT	13 kg

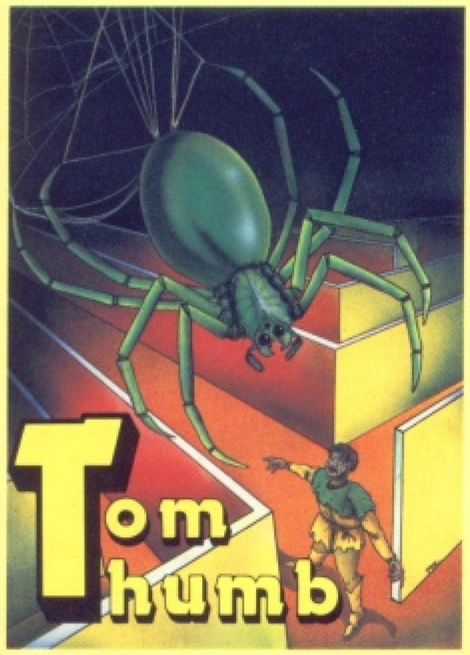
ANIROG

AT LAST 1 TAPE 2 MACHINES

COMMODORE 64 VIC 20







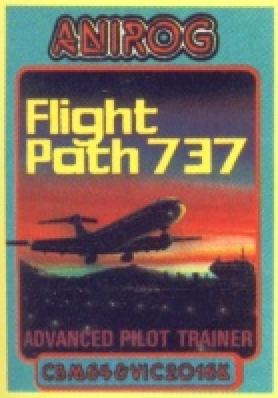
MINIPEDES

It is the height of summer and the garden is buzzing with bees and bugs. Minipede, a mutant mushroom monster advances relentlessly towards you, devouring everything in its path. 15 screens of fast and furious action make Minipedes a real challenge to the arcade enthusiasts.
Commodore 64 - VIC 20 16K
J.S. or K.B. £5.95

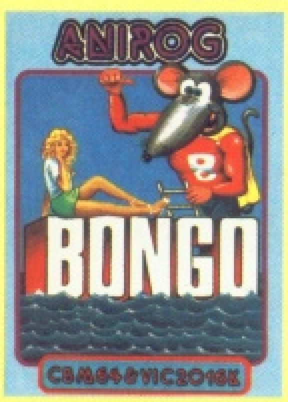
Experience the thrills of the gambling centre of the world from the comfort of your own armchair. Both versions include features such as spinning reels, hold, number-feature nudges, gamble/collect, spin score and hi-score. The Commodore 64 version has additional features, nudge, reward lucky 3, step-a-win and hi-score tables.
As with any arcade machine the odds are stacked against you!
Commodore 64 - VIC 20 16K **K.B. £5.95**

TOM THUMB

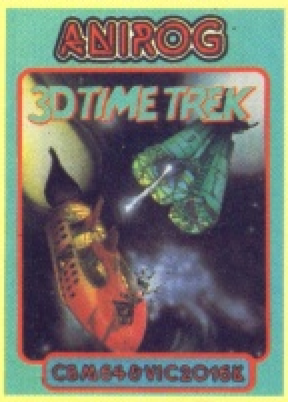
Tom is trapped in a scrolling maze populated by loathsome creatures, guardians of the lost treasures of the Magezam. Six separate screens, five levels of difficulty and four player option provide an exciting challenge for the whole family. Another stunner from the author of **BONGO!**
(1 to 4 players)
Commodore 64 - VIC 20 16K
J.S. £5.95



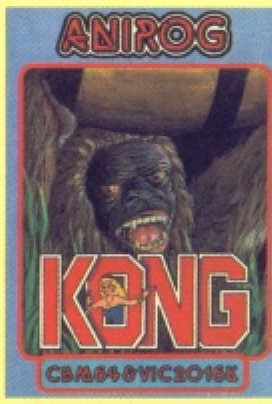
J.S. AND KEYBOARD
£7.95



J.S. **£7.95**



J.S. AND KEYBOARD
£5.95



J.S. or KEYBOARD
£7.95



J.S. **£7.95**

COMMODORE 64 VIC 20

TRADE ENQUIRIES: ANIROG SOFTWARE LTD. 29 WEST HILL DARTFORD KENT (0322) 92513/8
MAIL ORDER: 8 HIGH STREET HORLEY SURREY 24 HOUR CREDIT CARD SALES HORLEY (02934) 6083
PAYMENT BY CHEQUE P.O. ACCESS/VISA 50p POSTAGE & PACKAGING

BUSICALC 3

- the sophisticated spreadsheet !

Easy to learn, easy to use - something that can't be said of many business programs. But it's true of all the programs in the BUSICALC series.

BUSICALC 3 can handle all sorts of jobs - budgets, expenditure analysis, stock lists, price lists, and product costing are just a few of the possibilities. Three-dimensional formulae automatically access data stored on disk, so that you can easily pull together information from several different sheets and summarise or manipulate it.

It's simple to transfer data to other programs such as Easy Script. And you can use virtually any printer with BUSICALC 3, whether dot matrix or daisy wheel, Commodore or non-Commodore.

For the CBM 64 and PET/CBM 4000 & 8000 series.

Available through dealers or from:
Supersoft, Winchester House, Canning Road, Harrow HA3 7SJ

Phone 01-861 1166 for more details and a free catalogue.

